Holistic, data-driven, service and supply chain optimisation: linked optimisation.

OGUNSEMI, A.

2022

Holistic, Data Driven, Service and Supply Chain Optimisation

Linked Optimisation

Akinola Ogunsemi

The Robert Gordon University
School of Computing

Submitted in partial fulfilment of the requirements of the degree of
Doctor of Philosophy

July, 2022

Abstract

The intensity of competition and technological advancements in the business environment has made companies, as a means of survival, collaborate and cooperate together. This creates a chain of companies and business components with unified business objectives. However, managing the decision-making process (like scheduling, ordering, delivering, and allocating) at the various business components and maintaining a holistic objective is a huge business challenge as these operations are complex and dynamic. This is because the overall chain of business processes is widely distributed across all the supply chain participants; therefore, no individual collaborator has a complete overview of the processes. Increasingly, such decisions are automated and are strongly supported by optimisation algorithms - manufacturing optimisation, b2b ordering, financial trading, transportation scheduling and allocation. However, most of these algorithms do not incorporate the complexity associated with interacting decision-making systems like supply chains. It is well-known that decisions made at one point in supply chains can have significant consequences that ripple through linked production and transportation systems. Recently, global shocks to supply chains (COVID-19, climate change, blockage of the Suez Canal) have demonstrated the importance of these interdependencies and the need to create supply chains that are more resilient and have significantly reduced impact on the environment. Such interacting decision-making systems need to be considered through an optimisation process. However, the interactions between such decision-making systems are not modelled. So, we believe that modelling such interactions is an opportunity to provide computational extensions to current optimisation paradigms.

Therefore, this research study aims to develop a general framework for formulating and solving holistic data-driven optimisation problems in service and supply chains. This research achieved this aim and contributes by firstly, considering the complexities of supply chain problems from a linked problem perspective. This leads to developing a formalism for characterising linked optimisation problems as a model for supply chains. Secondly, the research adopts a method for creating a linked optimisation problem benchmark by linking existing classical benchmark sets. This involves using a mix of classical optimisation problems, typically relating to supply chain decision problems, to describe different modes of linkages in linked optimisation problems. Thirdly, several techniques for linking supply chain fragmented data have been proposed in the literature to identify data relationships. Therefore, this thesis explores some of these techniques and combines them in specific ways to improve the data discovery process. Lastly, many state-of-the-art algorithms have been explored in the literature, and these algorithms have been used to tackle problems relating to supply chain problems. This research, therefore, investigates the resilient state-of-the-art optimisation algorithms presented in the literature and then designs suitable algorithmic approaches inspired by the existing algorithms and the nature of problem linkages to address different problem linkages in supply chains. Considering research findings and future perspectives, research demonstrates the suitability of algorithms to different linked structures involving two sub-problems and this, therefore, suggests further investigations on issues

like the suitability of algorithms on more complex structures, benchmark methodologies, holistic goals and evaluation, process mining, game theory and dependency analysis.

This thesis is dedicated to my heavenly father for the gift of life.

Acknowledgements

I wish to express my appreciation to my supervisory team, Prof. John McCall, Dr Benjamin Lacroix, Dr David Corsar from Robert Gordon University (RGU), and Dr Mathias Kern and Dr Gilbert Owusu from British Telecommunications plc (BT). You showed enormous support and advice and you provided a high level of supervision. Most importantly, I am grateful to RGU, BT and TheDataLab for the generous funding, training and other resources provided to conduct my research. I wish to further thank RGU and National Subsea centre (NSC) for offering travel opportunities to participate in several events.

I wish to appreciate my fellow research students and staff members at the School of Computing and NSC for their support throughout the research journey.

Finally, I wish to say a big thank you to my family and friends for their support, love, encouragement and understanding.

Contents

List of Tables

List of Figures

Abbreviations

ALNS	Adaptive Large Neighborhood Search
ASHLO	Adaptive Simplified Human Learning Optimisation
AdvWork	Adventure Work
AHP	Analytic Hierarchy Process
ACO	Ant Colony Optimisation
ACP	Artificial Chemical Process
ACRO	Artificial Chemical Reaction Optimisation
AIS	Artificial Immune System
ATA	Artificial Tribe Algorithm
AP	Assignment Problem
BOA	Based Optimisation Algorithm
BOP	Bi-Level Optimisation Problem
B2B	Business-to-Business
CFO	Central Force Optimisation
CSS	Charged System Search
CRA	Chemical Reaction Algorithm
CRO	Chemical Reaction Optimisation
CS	Clonal Selection
CSA	Chemotherapy Science Algorithm
Col	Content-Based Similarity
CINM	Continuous Immune Network Models
CS	Cuckoo Search
Cosine	Cosine Similarity Approach
CEM	Cross Enthropy Method
DINM	Discrete Immune Network Models
DMS	Distributed Manufacturing System
DPFSP-NF	Distributed Permutation Flow Shop Scheduling Problem Non-identical Factory
DPFSP	Distributed Permutation Flow Shop Scheduling Problem
EMD	Earth Mover's Distance

EDI	Electronic Data Interchange
EAs	Evolutionary Algorithms
EP	Evolutionary Programming
ES	Evolutionary Strategies
EO	Extremal Optimisation
FLP	Facility Location Problem
FLPPFSP	Facility Location Problem and Permutation Flow Shop Scheduling Problem
FN	False Negative
FP	False Positive
FastFK	Fast Foreign Key
FOA	Fireworks Optimisation Algorithm
FIFO	First In First Out
FGA	Football Game Algorithm
FMINP	Fuzzy Mixed-Integer Nonlinear Programming
GBMO	Gases Brownian Motion Optimisation
GAMS	General Algebraic Modelling System
GDP	Generational Distance
GA	Genetic Algorithm
GP	Genetic Programming
GB	Golden Ball
GE	Grammatical Evolution
GSA	Gravitational Search Algorithm
GRAS	Greedy Randomised Adaptive Search
GHG	Green House Gas
GLS	Guilded Local Search
HUX	Half Uniform Crossover
HS	Harmony Search
HMODE	Heterogeneous Multi-Objective Differential Evolution Algorithm
HoPF	Holistic discovering of Primary Key and Foreign Key
HV	Hypervolume
ICA	Imperialist Competitive Algorithm
IOBSPR	Integrated of Order Batching, Batch Sequencung and Picker Routing
IGD	Inverted Generational Distance
IDP	Inventory Decision Problem
IDF	Inverse Document Frequency
IPM	Integer Polynomial Mutation
IWD	Intelligent Water Drops
ILS	Iterated Local Search

JAP	Job Assignment Problem
JAPTSP	Job Assignment Problem and Travelling Salesman Problem
KMMOA	Knowledge-Based Multi-Objective Memetic Optimisation Algorithm
KPIs	Key Performance Indicators
LCA	League Championship Algorithm
LPS	Linked Problem Systems
MS	Melody Search
MMC	Method of Musical Composition
MILP	Mixed-Integer Linear Programming
MIP	Mixed Integer Programming
MCDM	Multi-Criteria Decision Making
MCRGALP	Multi-Criteria Ranking Genetic Algorithm for Linked Problem
MMTSP	Multi depot Multiple Travelling Salesman Problem
MTSP	Multiple Travelling Salesman Problem
MULTIMOORA	Multiplicative form Multi-Objective Optimisation by Ratio Analysis
NSim	Name Similarity
NSA	Negative Selection Algorithm
NSI	Negative Social Impact
NSGA-II	Non-dominated Sorting Genetic Algorithm II
NSGALP	Non-dominated Sorting Genetic Algorithm for Linked Problem
PMX	Partially Mapped Crossover
POMSIC	Partial Optimisation Metaheuristic under Special Intensification Conditions
PSO	Particle Swarm Optimisation
PFSP	Permutation Flow Shop Scheduling Problem
PSM	Permutation Swap Mutation
PNLP	Possibilistic Non-linear Programming
PSA	Positive Selection Algorithm
PROMETHEE	Preference Ranking Organisation Method for Enrichment Evaluation
Pri	Pseudo-Primary Key Discovery
QAP	Quadratic Assignment Problem
RFID	Radio Frequency Identification
RHV	Relative Hypervolume
RFD	River Formation Dynamics
SS	Scatter Search
Sem	Semantic Similarity in a Taxonomy
SEQ	Sequential Approach
SOC	Service Oriented Computing
SCA	Sine Cosine Algorithm

SBX	Simulated Binary Crossover
SKFA	Simulated Kalman Filter Algorithm
Soundex	Soundex Similarity
SLCA	Soccer League Competitive Algorithm
SEOA	Social Emotional Optimisation Algorithm
SDS	Stochastic Diffusion Search
SMNLP	Stocastic Mixed-Integer Nonlinear Programming
SQL	Structured Query Language
SCM	Supply Chain Management
SI	Swarm Intelligence
TS	Tabu Search
TLBO	Teaching-Learning Based Optimisation
TOPSIS	Technique for Order Preference by Similarity to Ideal Analysis
TF	Term Frequency
TFIDF	Term Frequency-Inverse Document Frequency
TEC	Total Energy Consumption
TCPH	Transaction Processing Performance Council
TMP	Travelling Maintainer Problem
TSP	Travelling Salesman Problem
TTP	Travelling Thief Problem
TN	True Negative
TP	True Positive
TWO	Tug of War Algorithm
Usage	Usage-Based Approach
Val	Value Ranges Similarity
VNS	Variable Neighborhood Search
VRP	Vehicle Routing Problem
WPM	Weighted Product Method
WSM	Weighted Sum Method
WSRP	Workforce Scheduling Routing Problem

1

Introduction

This chapter introduces and presents the rationale for this research. It further presents the research aims and provides the research contributions as well as a summary of research publications undertaken during this research.

1.1 Overview

The intensity of competition and technological advancements in the business environment has made companies, as a means of survival, collaborate and cooperate together. This, therefore, creates a chain of companies and business components with unified business objectives [259]. However, managing the operations (i.e., scheduling, ordering, delivering, allocating) at the various business components and maintaining a holistic objective is a huge business challenge as these operations are complex and dynamic. This is because the overall chain of business processes is widely distributed across all the supply chain participants. Therefore, no individual collaborator has a complete overview of the processes [199]. Also, decisions made at one point in the supply chain can have significant consequences that ripple through linked production and transportation systems. Recently, global shocks to supply chains (COVID-19, climate change, blockage of the Suez canal) have demonstrated the importance of these interdependencies and the need to create supply chains that are more resilient and have significantly reduced impact on the environment.

It is, therefore, essential for the collaborating partners to access and maintain synchronised supply chain process models to identify the level of interactions between the participants in the supply chain [393]. This will facilitate the analysis of current collaborating processes such that existing processes can be efficiently and effectively optimised [199].

Due to the level of interactions and processes between several components in the supply chain network, participating organisations have developed a variety of heterogeneous data sources over time. As a result, data has become highly fragmented. However, processing contains several linked components, as evidenced by data managed by the individual organisation involved. While the data structure of each organisation may appear independent of each other, in reality, they are linked by a diverse set of relationships [116]. Let us con-

sider, for example, a supply chain decision problem involving a company operating a set of warehouses and a truckload trucking company operating a fleet of trucks, which the warehouse company contracts to fulfil demands at different retailers' locations. While the data regarding the demand fulfilment might be managed independently by both companies, the pieces of information regarding the locations of the warehouses, details of the demands, and the retailers' locations are available to both companies. These pieces of information are from the linkages between the warehouse data and the truck company data.

Data has become one of the most critical assets in the economy of the 21st century. Entire new industries rely on and are centred around exploiting large data sets. Many modern business processes generate millions or even billions of fragmented data records daily. In a real sense, several data sources are developed independently [7] and as a result, integrating the fragmented data from the individual components of a chain of business processes becomes more challenging. In addition, different data representations are designed and so, this makes the principles or semantic concepts dissimilar [202]. However, solving these challenges is key to business process modelling for optimising supply chain processes.

The concept of linked decision-making in service and supply management systems has increasingly attracted strong adoption of optimisation algorithms. Increasingly, decisions are automated or strongly supported by optimisation algorithms - manufacturing optimisation, b2b ordering, financial trading, transportation scheduling and allocation. However, most of these algorithms do not incorporate the complexity associated with interacting decision-making systems like supply chains [184].

Real-world problems, like supply chains, are systems characterised by such combination and interdependency, where some features of the components of the supply chain problem are linked [42]. A decision made at one point in the supply chain could have the significant consequence of rippling through linked production and transportation systems. This implies that an optimal solution for individual operational supply chain components might not guarantee an optimal solution for the overall supply chain problem [363].

Supply chains are increasingly complex and highly information-driven decision-making problems. Different units in the supply chain make decisions that have consequences for successors in the chain. We are interested in supply and service chains that can be modelled as linked optimisation problems, such that Unit A selects a solution to an optimisation problem. That solution has the consequence of creating an instance of an optimisation problem for Unit B, its successor in the chain. The term service chain refers to part of supply chain that focuses on providing services on products.

Moreover, many examples occur in the literature where particular linked problems are modelled, and evolutionary computation / computational intelligence solutions and mathematical modelling techniques are developed to solve the problem. While these are interesting examples, there is a lack of benchmarks available for the optimisation community to make a systematic study of linked optimisation problems and supply chains. Furthermore, a missing component in this analysis is the idea of a governing authority that has holis-

tic goals derived from the solutions adopted by different units across the chain. Holistic goals may range from corporate-level profitability to environmental goals such as net zero carbon. Our approach is to create an analytical and programmatic framework for linked optimisation problems so that supply chains can be studied systematically.

1.2 Rationale/Motivation

Studies have shown how organisations in the past make decisions on a single operational component of the supply chain without considering how other components are affected. As a result, they suffer considerable costs in satisfying their customers. Furthermore, it is said that a range of 3% to 20% of total operating cost could be reduced if all of the components in the supply chain are considered [68].

However, in today's dynamic business environment, solving the operational components of the supply chain as a whole is usually challenging and complex. This is due to the level of non-linearities and discontinuities, constraints, complexity and business rules, as well as conflicting objectives, noise and uncertainty among these components [266]. However, an optimal plan might be guaranteed that provides business success and customer satisfaction. Moreover, researchers have demonstrated that in practice, different modes of linked problems are evident in the real world and can be solved optimally depending on the modes of connection.

Similarly, linked problems are evident in the data structure [116], which contains key business information about resources, people, processes, events, technologies, transactions, and business relationships. In large organisations, for instance, numerous large databases are maintained that hold information about key business processes. These databases come in different formats, structures and data sources; however, expert knowledge about the interactions of these business processes is not easily available or accessible as information is highly fragmented and undocumented. The exploration of such data and relationships has mainly been addressed through highly time-intensive human analysis and exploration by domain experts [100] [109] [319]. Thus, the current approach becomes a tedious, time-consuming and error-prone process that is unsuitable in today's dynamic business environment [26][161].

In addition, due to the nature and the complexities of a chain of business processes. It is often challenging to determine the number of relationships in the data captured from the individual components of the business. However, capturing an appropriate level of abstraction of the processes in the data can guide us to finding a holistic model for formulating the optimisation problem across the supply chain network.

Therefore, the motivation for this thesis is to automatically infer from the data sets of relationships that exist due to the connectivity in the supply chain business processes. Thus, it is believed that this could form a prerequisite for effective and efficient holistic decision-making. To tackle the holistic supply chain optimisation problem, the thesis intend to ex-

plore and derive the properties of the linked components in the supply chain from data and use these properties to define and develop a formulation to support the holistic optimisation of the problem. Also, we intend to investigate approaches that automate the traditional tasks that domain experts employ for data analysis of supply chain processes. Technically, the research will be split into two major parts; linking fragmented supply chain data, and optimising the supply chain components from a linked optimisation perspective.

1.3 Research Aims

The primary purpose of this thesis is to research real-world problems in today's supply chain business environment, which consists of several interacting components. Therefore, the thesis aims to develop a general framework for formulating and solving holistic data-driven optimisation problems in service and supply chains. This will involve investigating algorithmic strategies that address local components under the framework of linked optimisation problem. In addition, this work addresses which strategies should be adopted when tackling the problem of different levels of complexity. It is important to note that our framework cannot reflect all the complexities of real-world problems in the supply chain. However, we aim to adopt methods that consider all the non-linear interactions and constraints in the problem model and find near optimum solutions.

1.3.1 Research Questions

Several research questions are defined below that this research attempts to answer.

(RQ1) How can we develop a formalism for the linked optimisation problem?

(RQ2) What are the various problem linkages in real-world supply chains?

(RQ3) How can data relationships be inferred and verified in a supply chain database model?

(RQ4) What algorithmic methodologies can be designed to tackle complex interactions in the supply chain network?

1.3.2 Research Objectives

We will be focusing our research on the following specific objectives.

(O1) To develop a formalism for linked optimisation problem.

This objective addresses (RQ1) in part. The research is keen to provide an understanding of the complexities of supply chain problems from a linked problem perspective. This will develop a formalism for characterising linked optimisation problems as a model for supply chains.

(O2) To identify and define various modes of problem linkages that occur in real-world supply chains.

This objective relates to the research question (RQ2). The research will adopt a method for creating linked optimisation problem benchmarks by linking existing classical benchmark sets. This will involve using a mix of classical optimisation problems, typically relating to supply chain decision-making problems, to describe different modes of linkages in linked optimisation problems.

(O3) To investigate a combination of approaches to link fragmented supply chain data.

This objective addresses (RQ3) in part. Several techniques for linking fragmented data have been proposed in the literature to identify data relationships. This thesis will, therefore, explore some of these techniques and combine them in a certain way to improve the data discovery process.

(O4) To develop and evaluate algorithmic approaches to address the linked optimisation problem.

This objective addresses (RQ4). Many state-of-the-art algorithms have been explored in the literature, and these algorithms have been used to tackle problems relating to supply chain problems. The research will investigate the powerful state-of-the-art optimisation algorithms presented in the literature. This will then lead to designing suitable algorithmic approaches inspired by the existing algorithms and the nature of problem linkages to address different problem linkages in supply chains.

1.4 Contributions

In this section, we present the original research generated in the process of meeting our objectives.

Contribution 1. Novel formalism for characterising linked optimisation problems. This allows systems of linked problems to be defined with a precise specification of linkages in terms of the solution domain, objective functions and constraints. The thesis further validates the formalism by applying it to several real-world examples extracted from this literature. This contribution addresses the research question (RQ1) and relates to the objective (O1). Research work is presented in Chapter 3 and is part of a journal paper planned to be submitted to a reputable journal paper.

Contribution 2. Identification of modes of linkages in real-world supply chains. The contribution provides insights into understanding linked problems in terms of network interactions. It explores a methodological framework for the linked optimisation problem and further presents several modes of linkages using a set of combined supply chain decision problems. This entails a method for creating benchmark problems by

linking existing classical benchmark sets. Contribution 2 addresses the research question (RQ2) and relates to the objective (O2). Research work is presented in Chapter 5 is part of the journal paper to be submitted to a reputable journal.

Contribution 3. Linkages of fragmented supply chain data. Contribution addresses the relational data discovery problem by investigating how eight data relationship discovery algorithms can be combined to identify potential links between supply chain data in different ways using different categories of database information. It proposes a voting system and hierarchical clustering ensemble methods to improve linkage discovery performance. Contribution addresses the research question (RQ3). In addition, the contribution is presented in Chapter 4 and has been published in Artificial intelligence XXXVII: proceedings of 40th SGAI Artificial intelligence international conference (AI 2020), 15-17 December 2020, Cambridge, UK.

Contribution 4. Application of three algorithmic methodologies to tackle two cases of supply chain problems from a linked optimisation problem perspective. Contribution entails the investigation of the linkages between a facility location and permutation flow shop scheduling problems of a distributed manufacturing system, and the integration between job assignment problem and travelling salesman problem of a service chain system where service personnel are required to perform tasks at different locations. This entails formulating a novel mathematical model from a linked optimisation perspective for each case study. The thesis presents three algorithmic approaches to tackling them; Non-dominated Sorting Genetic Algorithm for Linked Problem (NSGALP), Multi-Criteria Ranking Genetic Algorithm for Linked Problem (MCRGALP), and Sequential approach. Contribution addresses the research question (RQ4) relating to the objective (O4). Research work is presented in Chapter 6 and contribution leads to the publication of two conference papers. (1) Proceedings of the Genetic and Evolutionary Computation Conference Companion July 9-13, 2022, Pages 735–738, Boston, ACM, and (2) Artificial intelligence: proceedings of 42nd SGAI Artificial intelligence international conference (AI 2022), 13-15 December 2022, Cambridge, UK. In addition, the contribution is selected as the best referred application paper for the conference.

1.5 Publications

Section of the work presented herein has been published in the following peer-reviewed publications:

- Ogunsemi, A., McCall, J., Kern, M., Lacroix, B., Corsar, D. and Owusu, G., 2020, December. Ensemble-Based Relationship Discovery in Relational Databases. In Interna-

tional Conference on Innovative Techniques and Applications of Artificial Intelligence (pp. 286-300). Springer, Cham.

- Ogunsemi, A., McCall, J., Kern, M., Lacroix, B., Corsar, D. and Owusu, G., 2022, July. Facility location problem and permutation flow shop scheduling problem: a linked optimisation problem. In Proceedings of the Genetic and Evolutionary Computation Conference Companion (pp. 735-738).

- Ogunsemi, A., McCall, J., Kern, M., Lacroix, B., Corsar, D. and Owusu, G. Job Assignment Problem and Traveling Salesman Problem: A Linked Optimisation Problem. In International Conference on Innovative Techniques and Applications of Artificial Intelligence. To be published in December 2022.

1.6 Patents

This section presents two pending inventions filed with the intellectual property office:

- OGUNSEMI, A., MCCALL, J., KERN, M., LACROIX, B., CORSAR, D. and OWUSU, G. 2020.Inferring database relationships. U.K. patent application no. GB2002946.8. Intellectual Property Office.

- Ogunsemi, A., McCall, J., Kern, M., Lacroix, B., Corsar, D. and Owusu, G. Database Relationships Discovery. U.K. patent application no. GB2002947.6. Intellectual Property Office.

1.7 Organisation

The rest of the thesis is organised as follows: Chapter 2 reviews relevant works in the literature by focusing on real-world supply chain decision problems. In the chapter, we discuss issues relating to the different aspects of the supply chain in terms of data and lack of overall overview and then explore several supply chain techniques that have been presented in the literature. Chapter 3 develops a general formalism for linked optimisation problems and presents a validation of our formalism by applying it to several real-world examples extracted from this literature. Chapter 4 looks into an investigation of how several data relationship discovery algorithms for determining linkages in supply chain fragmented data. In Chapter 5, we present some insights into understanding linked problems in terms of network interactions and explore a methodological framework for linked optimisation problems using a mix of classical optimisation problems. Chapter 6 investigates two case studies by considering the linkages between a facility location and permutation flow shop scheduling problems of a distributed manufacturing system of a supply chain, and the integration between job assignment problem and travelling salesman problem. Lastly, Chapter

7 concludes the thesis and states future research directions. The chapter also discusses the limitations of approaches used in the thesis.

2

Basic Concepts and Literature Review

2.1 Introduction

The concept of supply chain management significantly improves the end-customers satisfaction [227]. However, managing a supply chain with different sources of uncertainty and where the bullwhip effect (i.e., a phenomenon that involves increased variability in orders as orders move upstream in the supply chain [371]) exists between the various entities of the supply chain, is challenging [23]. These challenges occur because the processes involved in a supply chain, such as scheduling, ordering, delivering, allocation, and many more, are executed by several parties. Therefore, no single party in the supply chain has a holistic overview of the activities executed [257] - the lack of holistic overview results in fragmented data and decision-making generated by such processes.

A typical supply chain network involves four basic entities (suppliers, manufacturer, distribution network, and customers) connected and influenced primarily through transportation, information sharing, and financial flows [259]. Due to limited information available to each party in the supply chain, there is a lack of explicit description of business processes; therefore, they are unavailable for analysis [257]. To understand what the supply chain network is all about, the following section explored the concept of supply chain management.

2.2 Supply Chain Management

Falcone et al. [127] define a supply chain as a network comprising facilities and distribution points that perform procurement of materials, the transformation of materials, and distribution of transformed materials to fulfil customers' orders. A supply chain exists in both service and manufacturing organisations [127]. Over the years, the intensity of competition and technological advancements in the business environment has compelled companies, as a means of survival, to collaborate and cooperate, hence, creating a chain of companies and business units with unified business objectives [259]. As a result, a new concept of organisational management became a significant business driver, hence, supply chain management (SCM). SCM covers a wide range of business functions combined with theoretical

domains, including system theory, logistics, inventory control, mathematical, optimisation, and computer modelling [182]. For successful organisational performance, the success of the supply chain in which the organisations participate is vital [391]. In other words, according to Heizer and Rende [168], developing long-term, strategic relationships with the supply chain participants can bring about effective supply chain management at strategic, tactical and operational levels that provide customers with maximum values. However, this is different in practice because business units operate interdependently, resulting in conflicting objectives among the participating supply chain units.

Several factors influence the efficiency of a supply chain and are essential for decision-making in a competitive market [127]. Examples of such factors include stock management, planning and scheduling, production costs as well as distribution strategies. These decisions in the supply chain are time-dependent and, therefore, can be classified into three categories; strategic (long-term), tactical (midterm), and operational (short-term) decisions [259]. Several papers have attempted to address the various classification of underlying challenges in supply chain management. However, the literature has not fully covered a complete classification of issues in a supply chain due to its wide range of concepts [345]. We adopt the classification list of challenges provided by Tako and Robinso [345] for a detailed classification. Figure 2.1 identifies a set of challenges faced in the supply chain based on the level of decisions taken.

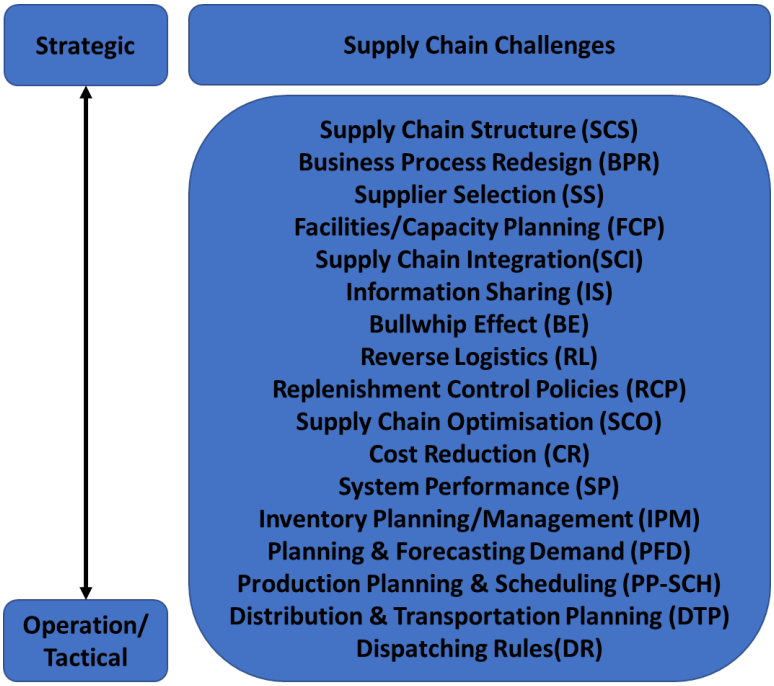


Figure 2.1: Supply Chain Issues Adapted from [345] p. 805

SCM provides an effective strategy that allows organisations to meet the challenges they face in current highly competitive and dynamic business environments [303]. The con-

nections among the individual participants in a supply chain form a complex structure whose individual behaviour affects the performance of the entire supply chain system [294]. Therefore, managing a complete supply chain requires a crucial competitive strategy that optimises decisions across several supply chain participants. This strategy must carefully consider the coordination and integration of all business operations across all boards and consider conflicting sustainability issues [403]. These considerations allow the evaluation of the quality of individual decisions that propagate from the components of the supply chain, which can be measured using various key performance indicators (KPIs) such as revenue, carbon emissions, makespan, and total distance travelled [182].

Assessing the likely outcomes of the given supply chain's decisions entails using a simulation. A computer simulates the supply chain's behaviour in the real world. The simulation is to find in reality the solution formed from a model of the supply chain problem [182]. Supply chain modelling allows us to clearly understand managing a supply chain as a simplification of a real-world supply chain. Two perspectives explain the significance of supply chain modelling: analysing supply chain dynamics to determine the strategies that minimise the effects of dynamic variation and validating a supply chain model that accurately represents the supply chain. We then use the model to generate a solution using a process of optimisation. Interestingly, both industry and academia have shown, to a large extent, much attention to supply chain optimisation. However, the configuration and optimisation of supply chain process remain a challenge at the strategic level [127].

There are many examples of supply chain modelling using optimisation algorithms in the literature. In Wang's study [368], the author considers the cost and benefit optimisation of a three-level supply chain network of suppliers, manufacturers, distribution centres, and retailers based on a genetic algorithm technique. The author defines a real-time inventory of multimedia products and establishes a bi-objective non-linear mixed integer programming for inventory estimation.

In the green supply chain management study of a used car resale company, Sathiya et al. [329] propose two strategies for cost and emissions reduction in a reverse supply chain network. The first strategy considers designing a mobile robot to reduce logistic costs and greenhouse gas (GHG) emissions. The next strategy adopts the concept of multi-objective optimisation to address the GHG problem based on Elitist Nondominated Sorting Genetic Algorithm (NSGA-II) and Heterogeneous Multi-Objective Differential Evolution algorithm (HMODE) [329].

Ghomi et al. [130] integrates production and distribution planning in a green supply chain and propose multi-objective particle swarm optimisation based on Pareto archive. The problem is formulated using a mixed integer programming model and seeks to maximise profit and minimise CO_2 emission. They conclude that integrating different components of a supply chain will enhance competitive advantages in terms of improved supply chain profitability.

Lotfi et al.'s study [244] addresses a closed-loop supply chain that accounts for sus-

tainability, resilience, robustness and risk aversion. They investigated a two-stage mixed-integer linear programming model and used the General Algebraic Modelling System (GAMS) software with CPLEX solver to minimise the costs, CO2 emission, and energy, along with maximising employment.

Ridwan et al. [315] propose a non-dominated sorting genetic algorithm II (NSGA-II) method to address the minimisation of operation cost and gas usage in the production of Cold Rolling Coil (CRC) and Cold Rolling Sheet (CRS).

Robles et al. [316] address a multi-objective hydrogen supply chain problem in terms of demand uncertainty. The authors explored a genetic algorithm approach based on a variant of NSGA-II multi-objective formulation to tackle the problem. In addition, a fuzzy approach was adopted for modelling demand uncertainty in hydrogen supply chain design. The paper considers three objective functions - hydrogen cost, global warming potential and safety risk index.

Shoja et al. [334] consider the minimisation of a two-stage supply chain network. Their study considers different transportation modes and the possibility of direct shipment between plants and customers. Modelling is based on a mixed-integer linear programming. The authors proposed thirteen classical and hybrid metaheuristic to solve the problem using an adaptive simplified human learning optimisation (ASHLO) algorithm, which is hybridised separately by genetic algorithm and particle swarm optimisation (PSO) algorithm. In addition, the authors adopt two additional meta-heuristic algorithms - gravitational search algorithm (GSA) and cockoo search (CS) for comparing performance.

The optimisation literature has expressed many such examples in each supply chain's specifics. Therefore, this thesis aims to abstract general patterns and principles in support of the academic study of supply chain optimisation problems. To this end, in the following section, we review some classical optimisation problems, instances of which are often identified as mapping to components of supply chain management problems.

2.3 Supply Chain Decision-Making Problems

The role of decision-making is to consider the values and preferences of decision-makers in identifying and choosing alternatives. Supply chain decision-making is primarily defined by complex and uncertain business environments driven by a large amount of data, decision variables, complex interrelationships among variables and system constraints, and performance trade-offs [253]. The most widely encountered decision-making problems in the supply chain include facility location-allocation, scheduling and planning, inventory, assignment, vehicle routing, travelling salesman, and knapsack decision. It is important to note that there is no intention to define an exhaustive list but instead to cover a good variety of classical problems to enable modelling a rich set of supply chain scenarios. These decision problems are ordered alphabetically in no order of relevance or importance.

2.3.1 Assignment Problem (AP)

The assignment problem is generally recognised from the 1955 publication of Kuhn's article in [224]. Assignment problem involves optimally matching the elements of two or more sets such that some objective function is minimised [293]. There are often two sets considered in most variations in the literature: "tasks" and "agents". Tasks are often referred to as jobs to be done, and agents are usually referred to as people or machines that perform the tasks. The classical version of the assignment problem matches each task to an agent, such that each agent performs at most one task (a one-to-one assignment). Other models discussed include; multi agents to a task assignment (many-to-one assignment) and multiple tasks to the same agent (a one-to-many assignment) [293]. See details in Table 2.1. AP is a real-world problem applicable in different job sectors; hospital resource planning – assigning the right resources/staff to patients, field Service Management – assigning maintenance tasks to technicians, transport/logistics/ supply chain – assigning set of delivery items/cargo to a finite number of trucks/ships.

Table 2.1: Variations of Assignment Problem

Model	**Variants**	**Description**
One-to-one Assignment	Classic assignment problem	The problem involves a one-to-one matching between n tasks and n agents and seeks to minimise the total cost of the assignments. Prime examples are assignment of jobs to machines, jobs to workers, or workers to machines [224].
	Classical assignment problem with agent qualification	The problem model is a variation of the classic assignment problem with m agents and n tasks. However, every agent may not be qualified to perform every task, and the objective is utility maximisation [60].
	The k-cardinality assignment problem	A variation of the classic assignment problem with m agents and n tasks, where only k of the agents and tasks are to be assigned, and k is less than both m and n [106].
	The bottleneck assignment problem	Unlike the classic assignment problem, the bottleneck seeks to find the sets of assignments that minimises the maximum value of the costs of assignments. An example is determining how to transport perishable goods from warehouses to markets without spoilage [156].

	The balanced assignment problem	Unlike the bottleneck assignment problem, the balanced assignment problem seeks to utilise both objectives by minimising the difference between the maximum and minimum assignment values [256].
	The minimum deviation assignment problem	The problem seeks to find a set of assignments that minimises the difference between the maximum and average assignment costs [159].
	The lexicographic bottleneck problem	The lexicographic bottleneck problem seeks to find the set of assignments that will not only minimise the largest cost coefficient in the solution but also minimise the next largest costs, without violating problem constraints [52].
	The $\sum_k$ - assignment problem	The objective is to find a set of assignments that minimises the sum of the k largest [157].
	The semi-assignment problem	This problem involves unique agents with identical tasks or vice versa. For example, a particular ship may require several radio operators with the same rank and skill level in the area of Navy personnel assignment [364].
	The categorised assignment problem	The problem divides the jobs into categories or groups, with sequencing requirements either within or between groups [301].
	Multi-criteria assignment problems	This problem variant involves multiple decision criteria, and it is important to find a solution that considers all the objectives [149] [231]
	The fractional assignment problem	problem involves non-linear programming, where the objective function to be optimised is the ratio of two other objective function expressions [333].
	The assignment problem with side constraints	The assignment problem with side constraints adds resource constraint or constraints to the classic assignment problem model [261] [139].
	The quadratic assignment problem	The quadratic assignment problem determines where to locate m facilities among n given sites, such that $n \geq m$ [172].
	The robust assignment problem	This class of assignment problem allows the decision maker to recognise uncertainty in the problem formulation [220].

One-to-Many Assignment (Multi-tasks)	The generalised assignment problem GAP	This is the most basic version of the assignment problem where an agent is assigned multiple tasks [64].
	The multiple resource GAP	This is a variant of GAP which considers how multiple resources may limit the agents' capacities [335].
	The bottleneck GAP	This problem model introduces a minimax objective for the GAP, which forms the Bottleneck Generalized Assignment Problem BGAP [260].
	The imbalanced time minimising assignment problem	This considers a variation of the bottleneck assignment problem where the number of agents m is less than the number of tasks n to be done such that every task must be done [11].
Many-to-One Assignment	The β-assignment problem	Unlike GAP, the problem is restricted so that only qualified agents can do certain jobs. There are three variations of β-assignment problem including; the cardinality β-assignment problem, the bottleneck β-assignment problem, and the weighted β-assignment problem [69].

2.3.2 Facility Location Problem (FLP)

Cooper in [93] proposed the FLP, which concerns the relation of a set of facilities to locations for delivering homogeneous services. Examples of such facilities include warehouses, production plants, distribution centres, and service centres. It plays a vital role in the strategic design of a supply chain network which concerns deciding on the best number and locations of the facilities subject to certain constraints. The literature has reviewed FLP from different perspectives [338], [265], [128].

FLP involves the selection of facilities to service a set of customers' demands at minimum cost. Cost includes the fixed cost of opening a facility and the cost of supplying customers' demand. FLP is defined by a matrix of costs of customer demands of size (no. of customers by no. of facilities), fixed cost of opening a facility, individual facility capacity, and quantity of customer demand. A facility can either be capacitated or otherwise. When a facility is capacitated, the total customers' demand assigned to each facility can not exceed the facility's capacity. Also, a customer's demand cannot be assigned to more than one facility. FLP has many applications, including; economics – locating businesses/service points, transport – freight terminals, emergency response, and locating waste collection centres within a waste collection network [99] [80].

2.3.3 Inventory Decision Problem (IDP)

Inventory means the number of goods and materials in stock [402]. The management of inventory adopts analytical processes that involves the optimisation of resources available for holding stock of several resources [44]. Inventory management is a significant part of core activities in a supply chain, and a lack of inventory can lead to stock-outs, resulting in a stalled production. However, high inventory can cause increased production costs due to high carrying inventory costs. Hence, inventory optimisation should ensure that stocks are not too low or too high; otherwise, this will have a negative impact on supply chain performance. Several decisions are required in inventory management; for example, in inventory policy, two basic decisions are required; when to replenish and how much to order. From a general perspective, the significance of inventory decisions is to identify the inventory policy that minimises the total cost of inventory, such as the ordering cost, carrying cost, and shortage cost. In a similar view, practical inventory decision problems should also address uncertainty, revenue, and deterioration [21] [358].

Most organisations are often faced with making decisions on large amount of individual items in relation to diverse factors (i.e., demand patterns, shipment and delivery methods) and constraints (i.e., budget limitations, supplier restrictions and different levels of customer service). These decision factors account for several dimension of inventory management design and have certain implications on inventory policies. For instance, Gutierrez and Vidal [160] reviewed inventory management models that are used in designing inventory policies in supply chains. These models are random demand models, random lead model, inventory policy models and integrated inventory management models. Zemzam et. al. [392] identified the techniques used in tackling inventory problem - analytical approach and simulation approach. Analytical approach uses a model to represent an inventory problem that combine decision variables with situational parameters with respect to the choice of inventory policy. On the other hand, simulation approach considers the testing of different scenarios and deciding on the best alternatives [392].

2.3.4 Knapsack Problem

The Knapsack Problem KP involves determining a set of items to be included in a knapsack with a given capacity (Weight) to maximise the total value of items in the knapsack. KP is a finite set of items with a given value and weight, the knapsack capacity. The problem's name is derived from a situation where a mountain climber has to select some items in his knapsack required for his trip. The problem has existed since 1897 and has been explored for more than a century [14]. KP is usually a sub-problem of more complex combinatorial optimisation problems [19]. The total assigned item weight must not exceed the knapsack capacity. KP has been applied to real-life problems like production and inventory management systems, cargo loading, yield management, airlines, and capital budgeting. Assi and Haraty [14] presented several variants of classical KP. They include; Bounded KP, Un-

bounded KP, Value independent KP, Multiple KP, 0-1 KP, Multidimensional multiple choice KP, Quadratic KP, and Online KP. Similarly, there are other sets of knapsack problems, including Nested knapsack, Collapsing knapsack, Non-linear knapsack, and Inverse-parametric knapsack [14].

2.3.5 Scheduling Problem

Scheduling involves determining the execution order of operations in a behavioural description [367]. A scheduling problem is defined by a triplet $\{\alpha, \beta, \gamma\}$. α describes the machine environment with a single entry. Processing characteristics and constraints are defined by β with no entry, a single entry, or multiple entries. γ defines the objective to minimise the problem, usually with a single entry. Scheduling is fundamental to sustaining competitive advantage of meeting customers' needs in many manufacturing and service sector. Planning and scheduling are regularly employed in decision processes, especially where processes are driven by communication, information processing, production, transportation, distribution and procurement [287]. In business organisations, completing activities requires scheduling functions to allocate limited resources to perform the activities. These scheduling functions are driven by mathematical and heuristic techniques [298] to optimise the organisation's objectives.

Different types of scheduling have been explored in the literature. The main types include; static and dynamic. Static scheduling has a fixed schedule structure which does not change over time. An example is seen in the seasonal bus schedule plan - the dynamic schedule structure changes with time [395]. Another example could be the resource scheduling of field engineers of a telecommunication company [365]. Furthermore, past literature has shown different classes of scheduling. These can be divided into two categories, as seen in Figure 2.2 - work scheduling and service system scheduling.

In scheduling problem papers, authors have explored several variations of scheduling. We provide the variations below

- Flow shop involves m machines where each machine must process each job. All jobs must follow the same execution plan on the machine. A job must join the next queue at the next machine after completion on one machine. If a First In First Out (FIFO) approach is maintained for job queues, the flow shop is referred to as a permutation flow shop scheduling problem (PFSP) [298]. PFSP schedules a set of jobs on a set of machines to minimise makespan [20] [407]. PFSP has essential applications in manufacturing and service systems, and the makespan typically represents the economic criterion in the shop scheduling problems [373].

- Flexible flow shop is a variant of a flow shop operating under parallel machine environments. It involves a series of stages where each simultaneously operates many

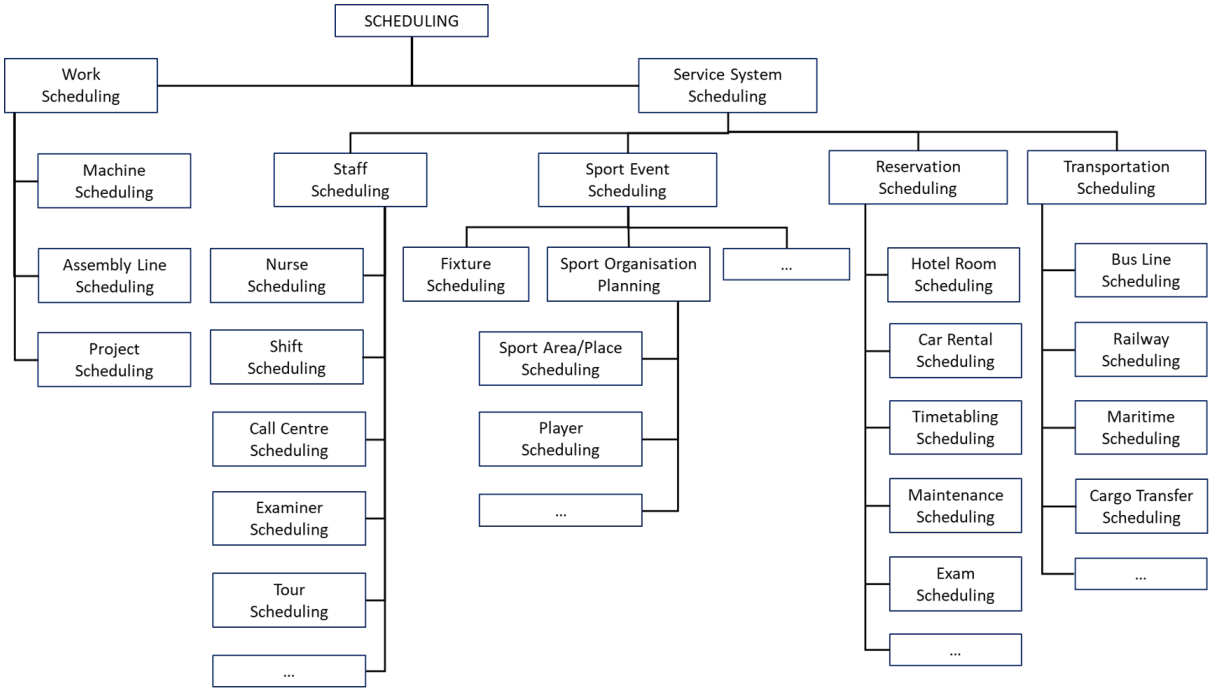


Figure 2.2: Scheduling Categories Adapted from [287] p. 1698

identical machines. Each job is executed in a specific order of stages. This flexible flow shop is known in the literature as a hybrid flow shop or multi-processors.

- Job shop with m machines predetermines the route that each job will follow. In job shops, each job may visit each machine more than once.
- Flexible job shop is a generalisation of the job shop with parallel machine environments. A flexible job shop takes a series of work centres where each work centre contains several machines in parallel. Here, each job has its route to follow through the shop. A job going through the shop may visit a work centre more than once.
- Open shop involves executing each job on each machine where some processing times may be zero. Also, there are no restrictions on how each job routes through the machine environment. Hence, different jobs may have different routes.
- Other variations of scheduling problems introduce several constraints, including due dates, release dates, preemptions, precedence constraints, machine breakdowns, blocking, machine eligibility restrictions, and batching [298].

2.3.6 Travelling Salesman Problem

The travelling salesman problem (TSP) is one of the famous classical optimisation problems that involves determining a tour of a salesman that minimises the total travel distance [151]. Over the years, there have been investigations of different formulations of the travelling

salesman problem. Investigations extend to formulations that include various transportation scheduling problems, like the multi-travelling salesman problem, the delivery problem, the school bus problem, and the dial-a-bus problem [147].

[201]

2.3.7 Vehicle Routing Problem (VRP)

One important factor in the supply chain is the physical flow of materials and goods, and if this is efficiently managed, it creates value. Vehicle routing decisions involve determining effective transportation of materials and goods to minimise total cost. Each trip starts from a depot to customers and ends at the depot. In addition, each client is only visited once. The total demand handled by individual vehicles must not exceed the capacity of the vehicle [86]. VRP is a generalisation of the TSP and was first introduced in [97]. TSP considers a single vehicle that visits multiple cities, whereas, the VRP considers multiple TSPs [350]. There are several variants proposed in the literature [402]. Eksioglu et al. [121] presents a methodology for classifying the literature on vehicle routing problems. These variants include; classical vehicle routing problem [97], vehicle routing problems with loading constraints [186], vehicle routing problem with time windows [37], stochastic vehicle routing problems [340], inventory routing problem [57], a multi-depot vehicle routing problem [173], pickup and delivery problems [9] [94] [288] [295]. Pillac et al. [297] present a survey on dynamic vehicle routing by classifying vehicle routing from the perspective of quality and evolution. [239] presents a review of green vehicle routing problems on the premises of energy consumption, emissions, and reverse logistics.

2.4 Real-World Optimisation Problems in Supply Chains

2.4.1 Linked Optimisation Problem

A real-world supply chain exhibits linkages between the individual operating components such that an optimal solution for a given component might not guarantee an optimal solution for the overall problem [363]. In other words, business decision problems are made up of interconnected components [341]. However, solving these problems to optimality requires a holistic approach. Real-world problems are systems characterised by combination and interdependency, where some features of the different components of the problems are linked [42]. In that sense, one of the attributes that make real-world problems more complex is the fact that the underlying problems are dynamic [184].

Assuming two components of a real-world problem are to be linked, a message (solution) is transmitted between the components where the solution of an individual component determines the quality of the other component by affecting the features of the receiving component. The problem can be understood from the concept of a dynamic optimisation

problem, which is characterised by three definitive properties including; time-varying objective function, time-varying input variables and time-varying constraints. Similarly, these properties can be extended to the linked optimisation problem since the sources of the dynamic complexity of systems are reflected in the properties of the linkages themselves. We can, therefore, classify these characteristics into three categories; link induced-varying objective function, link induced-varying solution set, and link induced-varying constraints. A linked problem can contain a single or a combination of these characteristics.

In the literature, there are many examples of optimisation problems where links between problems have been identified and solved from a linked perspective. In such studies, the solution of one problem dictates the outcome of others. A practical example is identified in [341], where a decision support optimisation system is used for a complex logistics operation involving five connected problems; facility location, truck scheduling problem, packing problem, driver selection and vehicle routing. Another instance of linked optimisation problem is found in Chen et al. [73], where authors identified significant interrelationship between crane handling and truck transportation and tackled the problems by integrating them using linked measures. Similarly, Chen et al. [75] investigated three problems that involve the integration of order batching, batch sequencing and picker routing (IOBSPR) in warehouses. They proposed an algorithm linking hybrid-coded genetic algorithm and ant colony optimization to tackle the IOBSPR model. The proposed strategies for typical linked problems identified in the literature are mostly solved separately and sequentially and this has often led to suboptimal joint solution [274]. It can be argued that linking interdependent optimisation problems can result in efficiency improvements and cost savings which, on average, can lead to 5-20% of cost savings [274]. To the best of our knowledge, no study has yet provided an explicit unified model that defines the features of linked problems and how such components affect each other. So, the abstraction of the linkages in a supply chain is worth addressing. The detailed formalism of the linked optimisation problem is provided in Chapter 3.

A succinct example of a linked optimisation problem is a multi-modal transportation business process that facilitates supply chain integration. Figure 2.3 depicts a multi-modal transportation process where every participant is involved in a predefined coordination of the main transportation process. The transportation process links each member in the supply chain network such that any deviation in the predetermined transportation process may impact the delivery of goods to the buyer. This is because the overall transportation process is not known to any party and therefore, any decisions made by one partner will not take into consideration of the overall transportation network.

These linkages are evident in the data structure as most real-world data are not independent but exhibit diverse relationships [116]. These relationships, in context, should provide us with an essential understanding of the properties of the problems, which are, in their own right, worthy of analysis. Having this in mind, identifying and understanding the different linkages is imperative in developing and applying different approaches to solving them.

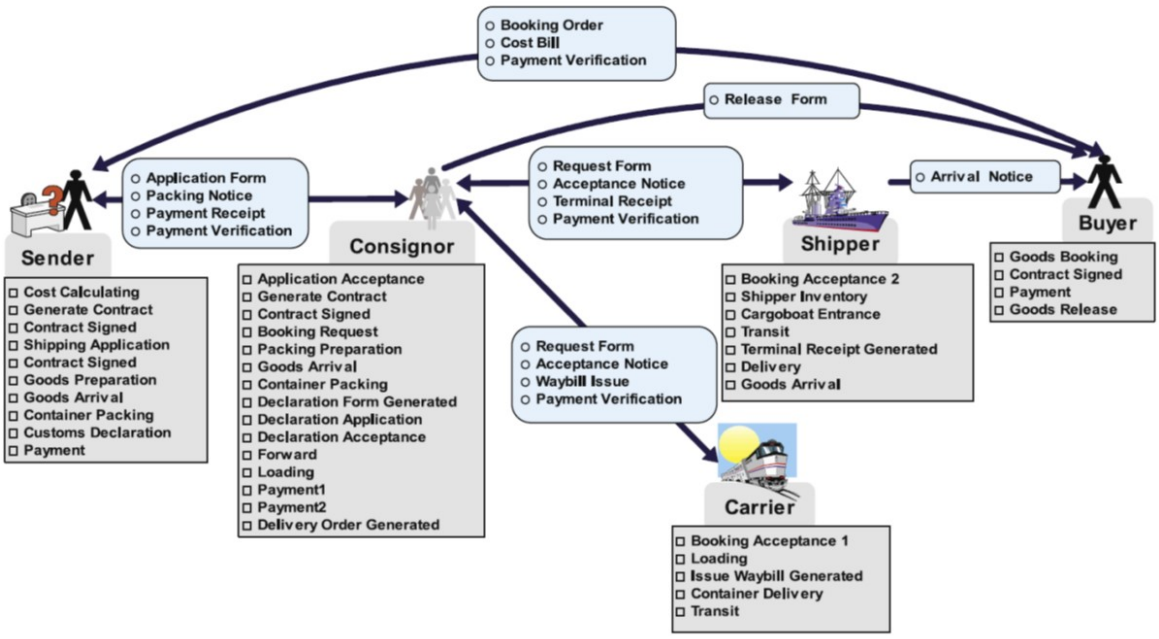


Figure 2.3: Multi-modal Transportation Process [393] p. 1282

To understand and address the challenges of linkages in supply chain optimisation problems, we will need to explore the issues that have posed many challenges to tackle. These issues are considered in three phases: the challenge of linking fragmented data that propagates from the decision points in the supply chain data, the challenge of mining the business processes to understand the process sequence in the supply chain, and problem with using appropriate computational techniques for tackling the supply chain decision problems.

2.5 Fragmented Supply Chain Data

The interoperability between the components of a chain of business processes results in the generation of tens or hundreds of millions of data records. These pieces of data contain event information which is stored in unstructured form. The extraction of such data forms a significant aspect of the optimisation of linked problems because problem linkages are evident in the data generated due to processes that are executed at different levels of the problem. However, data analysis and processing are complex and challenging due to the volume and complexity of data [321]. In most cases, they require some effort to extract as they are highly fragmented over many tables in a database model [357].

2.5.1 Linking Fragmented Supply Chain Data in a Relational Database Model

Linking fragmented data in the supply chain domain involves integrating data from different components of a supply chain system. Fragmented data occurs due to business in-

teractions between supply chain participants. The interactions involve several processes, resulting in hundreds or thousands of tables and attributes in a database. Hence, it is often challenging to determine the number of relationships in the data captured across the supply chain network as this would require matching unfamiliar data sources.

Matching unfamiliar data sources is a substantial burden [39] as this often requires the availability of domain experts that understand these sets of data. In a well-designed database, the type of information stored and the interrelationships of different data structures are well understood and well documented, but this is not always the case in real-world supply chain scenarios. In most cases, domain experts may not be readily available for several reasons.

Prior work in this area includes schema matching, which has extensively been studied [122]. Schema matching typically involves the manipulation of database schema elements to produce a mapping between elements of database schemas that are semantically related to each other [306]. Schema matching is vital in integrating data from heterogeneous sources where massive amount of data is produced and stored. Current practices in relational databases use multiple databases to store information due to size constraints [262]. These databases come in different formats and structures and require data integration to facilitate effective and efficient holistic decision-making.

However, understanding specific database schemas is often a time-consuming, and costly process [164]. This problem occurs because the information in several complex databases is interconnected, but these connections are often not obvious and easy to extract by users. A much more exhaustive approach is carried out manually in determining these connections, which is often done at the row level. Therefore, there is a need to automatically determine these connections for data integration [163].

Alwan et al. [7] identified four levels of schema information exploited in solving schema matching problems. These include: schema level information, instance level information, hybrid level information and auxiliary level information.

Schema level information explores three sub-level information to solve a schema matching problem; 1. linguistic level, 2. constraints level and 3. structure level. At linguistic level, meta-data information such as attributes' names, textual descriptions [115] and abbreviations and acronyms [310] are used in identifying relationships between schemas. Constraint level information exploits data types, value ranges and key types while structure level uses schema structure and schema attribute cardinalities [7] [306].

At the instance level, information is derived from the content of the schema such as values and rows/instances [132] [100] [202]. Such information is used in many cases when it is difficult to obtain sufficient information about the schema structure for matching, especially in real-world situations. In such cases, data instances may be available in determining relationships between schemas [7].

Hybrid level information combines both schema level and instance level information. For example, Li and Clifton [233] utilise the schema design information (schema and con-

straints) and data contents (data patterns and statistics) for semantic integration of schemas. Also, Zhao and Ram [401] exploit schema elements information in terms of names, descriptions, schema specification, data and usage patterns combined with data instances to validate relationships between schemas from different heterogeneous sources.

Auxiliary level information relies not only on existing schema information but also uses external information such as WordNet/Thesauri and dictionaries to determine the relationship between schema attribute names with the same contextual meaning [7]. For example, auxiliary information like thesaurus was used in expanding abbreviations of schema attribute names to determine attribute names that are semantically similar [249].

Recent work by Elmeleegy et al. [122] exploited another level of information based on usage information of schema attributes in query logs. Similarly, Ding et al. [108] exploited information from query logs based on usage statistics to find relationships between schema attributes to be matched. Usage information of attributes in queries may be needed when there is no sufficient schema information or data instances.

2.5.2 Schema-Based Matching

Schema-based matching is an important research topic in the Artificial Intelligence (AI) community due to the level of understanding required in natural language processing [109]. Thus, several studies have been geared towards speeding up the schema matching process using different AI techniques as opposed to the traditional labour-intensive procedure performed by domain experts.

Several researchers have explored the schema-based matching problem for specific or generic applications [112]. Table 2.2 describes approaches proposed in the literature for schema-based matching. The table also includes information levels used in matching several schemas.

Table 2.2: Schema-Based Matching Approaches [7] [30] [112]

Approaches	**Description**	**Matching Information**
Linguistic Matching	This technique implements algorithms to exploit meta-data information to find the relationships between tables based on attribute names and descriptions. It utilises tokenisation, stemming, string and substrings matching and information retrieval techniques.	Attributes' names [249] [114] [61] [112] [319] [109], Abbreviations/ Auxilliary Information [61] [114] [249] [112] [310] & Attributes' Descriptions [115]

Constraint-Based Matching	These techniques are algorithms that deal with the internal constraints applied to schemas definitions. Schemas contain constraints information such as datatypes, value ranges, uniqueness, optionality, relationship types, and cardinalities. Such information can be used to determine the similarity of schema elements.	Datatypes [234] [249], Referential-Constraints [61] [249] & Data Length [234] [319] [249]
Structure-Based Matching	Algorithms inspired by this approach use matching combinations of elements that appear together in a structure. The approach concentrates on the structural information and utilises constraints information of the targeted schemas to extract similarity between the attributes.	Keys [249] & Index [118]
Usage-Based Matching	Exploit usage information of the attributes in database objects. Usage-based schema matching identifies co-occurrences patterns between attributes and features like joins and aggregate functions.	Database query Logs [122] [108] & Key-word query logs [280]
Machine learning	It involves exploiting machine learning techniques to automate schema matchine.	Bayesian learning of domain expert examples to find optimal matching [28] & Clustering techniques for multiple schema attributes [109]

2.5.3 Instance-Based Matching

Instance-based matching employs data contents as a source to identify the relationship between schema attributes [7]. Instance-based matching is often used whenever it is impossible to use the schema information to perform an accurate match between schemas. The identification of instance-based correspondence could be fed back to the schema-based to improve the understanding of schema-based correspondence [401].

Instance-based matching has been used under different names by different research communities. In the database community, instance based matching also means instance identification [370], entity idenfication [143], merge purge [170], approximate record matching [361], [372], and record linkage or record matching [131], [360]. In the artificial intelligence community, instance matching is also referred to as database hardening [87] and name matching [33].

Most techniques used in instance-based matching are similarity metrics, and they can

be categorised into character-based matching (e.g. Ngram, Jaro-Winkler) and token-based matching (e.g. Jaccard) [262]. However, several approaches have been proposed to mitigate the weaknesses peculiar to the individual similarity metrics. Alwan et al. [7] identified four main strategies that exploit the contents of the database for instance-based matching problems. They include neural network, machine learning, information theoretic, and rule-based [7]. These approaches are shown in Table 2.3 below.

Table 2.3: Instance-Based Matching Approaches [7] [262]

Approaches	**Description**	**Matching Information**
Neural Network	This technique finds instance relationships by generating the similarities between data and predicts solutions from data with no knowledge about regularities.	Characteristics of data distribution [237] & Data pattern [381]
Machine learning	This approach adopts machine learning methods like Naive Bayesian classification to produce accurate matching results based on schema information. Approach usually uses a training data set to derive most appropriate matches.	Classification of data contents [317] [115] [196] [132] & Clustering based on data distributions [397]
Information theoretic	Approach is based on mutual information and distribution values to identify instance correspondences.	Data contents [238] [278]
Rule-Based Matching	This approach uses matching rules defined by first-order logic.	Use of schema information, summary instance information and instance properties of attributes [83] & Data values/ instances [34] [81]

2.5.4 Hybrid-Based Matching

Hybrid-based matching combines several matching approaches as a single approach by using multiple criteria and different information sources to predict a more accurate match result [7]. Some examples of hybrid-based matching are provided in Table 2.4 below.

Table 2.4: Hybrid-Based Matching Approaches [7] [263]

Approaches	Description	Matching Information
Rule-Based matching and regular expression	This approach combines set of rules and regular expression to determine matching patterns of of instances. The approach utilises a predefined regular expression and then apply a set of rules to classify schema attributes. The regular expressions are predetermined using statistical data and are fully explored for schema matching.	Attribute instance patterns based on a predefined regular expression [388]
Neural networks and rule-based	Approaches are adopted based on two-step processes. Neural network analyse data relationship pattern and then apply rules to filter the candidate pairs to generate the correct matching result.	Data pattern [381] & Schema and instance information [233]
Combined Machine learning (Unsupervised and Supervised learning)	This approach combines unsupervised cluster analysis techniques and supervised classification techniques to identify matching tuples.	Schema level information and instance-level data [401] & Classification of inclusion dependencies based on values and schema elements [319]
Regular Expression and Google Similarity	Approach combines the strength of regular expression approach as pattern recognition and google as web semantic to discover correspondences between schema attributes	Fully exploits data instances [263]

2.5.5 Composite-Based Matching

Composite schema matching involves the combination of outcomes obtained from different approaches used independently in matching database schemas [7]. In other words, a framework that combines the results of several approaches can provide a more comprehensive and better discovery of the relationships between database schemas than a single approach which may lead to imperfect matching of database schemas [306]. In composite-based matching, aggregation such as maximum, minimum, average, weighted sum, etc., are often used in obtaining an integrated result [386].

A prominent research study in the area of composite schema matching is seen in [113]. They propose a generic schema matching tool, Coma++, by combining different matching results. Their approach support schema-based and instance-level matching, utilising linguistic, structure and instance-based approaches. Coma++ offers a comprehensive frame-

work for large and complex schemas where different match strategies can be applied. Their results can be refined using different aggregations such as the minimum, maximum, average and weighted sum. Also, Yu et al. [386] proposed an approach based on the concept of composite schema matching. They combined linguistic similarity, path similarity, structure similarity and knowledge (semantic) similarity using a weighted sum to aggregate the similarity values from the individual matchings.

As part of our research contributions, we intend to explore different strategies by which we can combine the prediction of several techniques into an overall framework. According to Alwan et al.'s claims in [7], composite is more flexible than an individual or hybrid-based matching because the composite approach exploits the application domain and input schema information, which uses different approaches at different levels.

2.6 Mining and Learning Supply Chain Processes

The issues with uncertainty and complicated supply chain architecture have significantly influenced the need for supply chain partners to maintain synchronised process models for supply chain-wide processes. Process models can deliver an efficient and effective analysis of collaborative processes within the supply chain [199]. As a result, in accessing and maintaining these processes, several intelligent logistics information systems are implemented by collaborating partners to support decision-making in areas such as production scheduling, physical distribution, and material handling [210]. However, partial information about the business processes is captured by the information system of the individual supply chain collaborators where, in most cases, each partner does not have an insight into the overall process [257].

Moreover, according to Maruster et al's [257] assertion, partial information captured by the individual information systems of a chain of business partners can be combined to infer an overall distributed process, provided that there is a common reference point for each supply chain partner. This is because the components in the supply chain network are closely related such that a slight change in one component can significantly affect the other logistic operations of different components [213]. As stated by [227], attributes in one aspect of the supply chain network interact with elements in the other levels of the supply chain. In other words, obtaining an effective process formulation for the entire supply chain network may fail if one pays too much attention to one aspect of the supply chain.

In the context of supply chain process modelling, process mining techniques have been employed to develop an overall process model [199]. This technique can reveal the overall business processes within a supply chain network [257]. Process mining techniques involve a wide range of automated techniques that study processes based on historical process data [227]. In other words, process mining involves discovering, monitoring and improving busi-

ness processes based on the knowledge extracted from event logs available from information systems [357].

The most widely used techniques in business process mining are based on process discovery [84] and these techniques provide a complete picture of an entire business process model [199]. These techniques are often based on association rules and association clustering. For instance, [227] proposed a process mining system based on fuzzy association rules for knowledge discovery in a supply chain network using daily captured logistics operational data. Their model is shown in Figure 2.4.

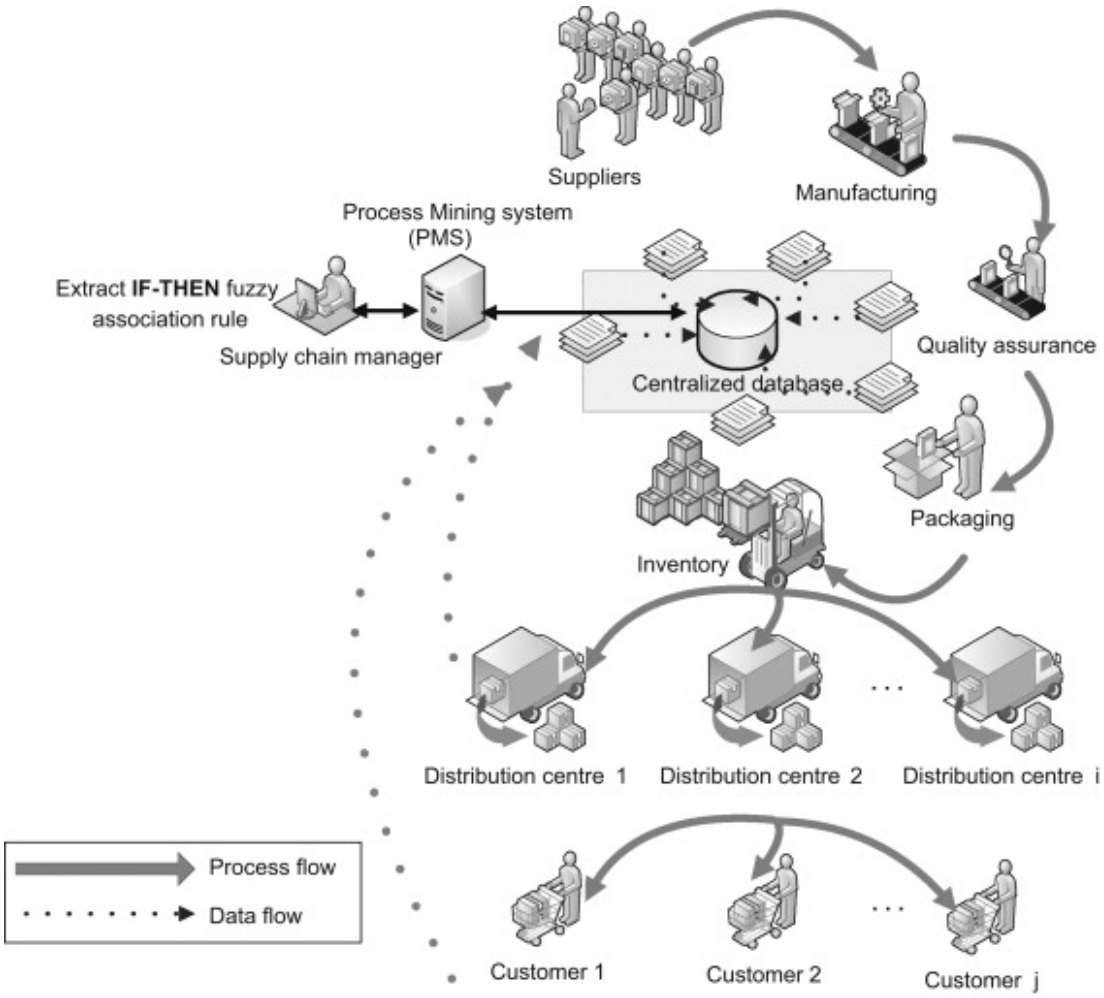


Figure 2.4: Supply Chain Process Flow [227] p. 177

2.6.1 Process Mining Techniques

Limited research work is proposed in the area of supply chain process mining. However, as noted in [199], the majority of the existing research papers focus on two techniques; data preparation (i.e. construction of an event log from historical process data) and process discovery (i.e. construction of process from an event log). We present in Figure 2.5, Van Der

Aalst's [357] process mining framework and in Table 2.5, a description of individual techniques in Van Der Aalst's framework.

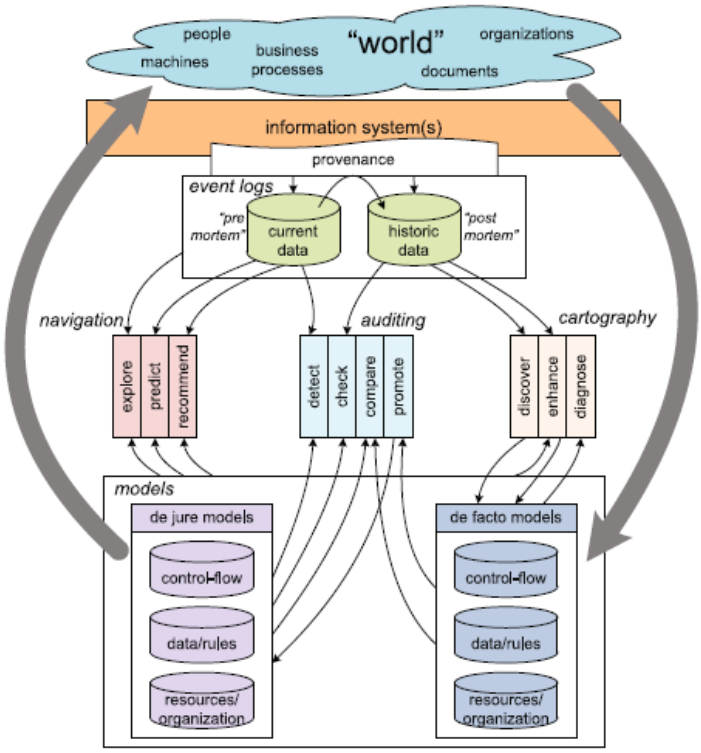


Figure 2.5: Process Mining Framework by Van Der Aalst [357]

In Figure 2.5, the term "world" represents an ecosystem of people, organisations, technologies, and business processes. The data that propagates from this ecosystem is recorded in the system(s) of information where event logs are captured systematically and reliably based on the term *provenance*. Provenance explains any form of information that describes the production process of an end product, which can be anything from a piece of data to a physical object [171]. Based on Figure 2.5, two types of event data are captured. Pre-mortem (i.e. current event data) and Post-mortem (i.e. historical process data). After an event has elapsed, the event data becomes a historic data and that is moved from the current data platform to the historical database. Process models are developed based on ten process mining activities, which can be grouped into three categories; navigation, auditing and cartography [357]. Note that provenance is also considered a process mining activity/technique. However, it cannot construct a model in its own right compared to other activities. Two models were identified by Van Der Aalst [357] in the framework; de jure and de facto. De facto process models are derived from reality, while de jure process models are designed to influence reality.

Table 2.5: Description of Process Mining Techniques [357]

Techniques	**Description**
Provenance	Focuses on systematic collection of process information required for constructing event log.
Discovery	technique uses an event log to construct a process model without using any knowledge of the existing process model.
Enhance	technique uses information about the actual process available in the event log to improve an existing process model.
Detect	A conformance checking operational support activity that compares a de jure model (i.e. an ideal model) with current event data (i.e. running process instances) to identify whether there are deviations at runtime. it generates an alert in the event of a violation of predefined rules.
Check	This operational support activity checks historical event data with an ideal process model (de jure model) to identify deviations in order to quantify compliance level.
Compare	This compares an ideal model (de jure model) with an actual model (de facto model) to determine whether there are deviations from what was planned or expected. This technique does not use event logs directly.
Promote	This approach uses the analysis between an ideal model and an actual model to improve existing processes based on proven best practices to update and promote an ideal model.
Explore	technique combines event data and models to compare and visualise running cases of events with earlier executed cases that are similar.
Predict	An operational support activity that makes statements about events based on historical event data where the future case is unknown.
Recommend	An operational support activity that learns from historical event data to provide a statement about a set of possible actions from a decision space.

In Table 2.6, we present Jokonowo et al.'s [199] classification of process mining techniques proposed in the literature. These techniques are based on Van Der Aalst's [357] framework. The table includes the authors' names, the proposed approach/methods used, the research focus, and the particular techniques being addressed.

Table 2.6: Supply Chain Process Mining Techniques adapted from [199] [357] [394]

Models/Methods	**Research focus**	**Techniques**
A process mining algorithm and a prototype implementation that discovers a set of fuzzy association rules for inter-organisational dependencies based on the daily captured logistics operation data [227].	Process mining for knowledge discovery	Discovery
A prototype implementation of an algorithm that generates event logs from RFID data based on product codes [152].	Processing Radio Frequency Identification (RFID) events for supply chain process mining	Provenance
A framework for extracting process data from application systems and integration portals like SAP enterprise software [211].	Mining of processes in Service Oriented Computing (SOC) environment data	Provenance
An approach for cross-correlation of process models and actual observed process behaviour across organisations based on some process analysis metrics [49].	Process mining analysis in service-based and cloud computing environments.	Check, Compare, Promote and Recommend
A correlation algorithm that transforms observed inter-organisational messages to an event log to associate individual messages to process instance [124].	Process models from Electronic Data Interchange (EDI) messages	Provenance
A methodology, system architecture and implementation of end-to-end business process insight and collaborative analytics platform for business users [321].	Leveraging process intelligence and process-aware analytics	Provenance, Discovery, Compare and Predict
A methodology designed to implement consistent process mining in a Big Data context [16].	Integrating multiple data sources using Big Data techniques	Provenance
An approach for supply chain process based on Ant-colony optimisation ACO that captures and optimises monitoring requirements to enact an optimal monitoring process through a service-oriented approach [90].	Service-oriented business process mining for monitoring enactment.	Recommend

An approach to discover the coordination patterns between different organisation and workflow models using information about resource allocation from running logs. A process integration approach that uses the mined process model for each organisation and the coordination patterns to obtain a cross-organisational workflow model [393].	Application of process mining for workflow integration using the concept of Petri net with comprehensive resource and message factors.	Discovery
An approach that applies online process mining techniques to cloud-based infrastructure for the runtime extraction of business rules from event logs [29].	Cloud computing architecture to support cross-organisational process executions	Discovery
A method and a rule-based algorithm that searches for links between data and merges data from the different inter-organisational processes for process mining [85].	Merging event logs for process mining.	Provenance
A privacy-preserving business process recommendation and composition system based on process mining and classification techniques for executable business process workflow regeneration [188].	Cloud-based privacy-preserving process mining in supply chain environment.	Discovery and Recommend
An approach and software implementation that integrates EDI, process mining, Business Intelligence and Semantic technologies for analysing and discovering inter-organisational process models [123].	Process models from Electronic Data Interchange (EDI) messages	Provenance and Discovery
A privacy-preservation cross organisational business process mining framework that handles privacy issues for process discovery [243].	Privacy-preservation in process mining	Discovery
A methodology that creates an enhanced model with an improvement of performance level based on Petri nets and events logs [104].	Process mining for process improvement	Check and Enhance
An investigation of the characteristics of event logs in make-to-order production analysis [290].	Performance analysis of workload analysis and delay analysis in the manufacturing processes.	Enhance

A method based on comparative case clustering robust with changes in behaviours [176].	Detecting changing behaviour in real-life business processes.	Detect
A method proposed to guide the analysis of capacity usage based on recorded data about railway operations [192].	Detecting train reroutings.	Detect
A method that extends the a multi-agent-based business simulator with resource event ontology [190].	Exploitation of the characteristics of a trading company to verify and explore relationships among agents.	Provenance, Check and Discovery
A method that utilises multidimensional process mining to map physical logistics activities [215].	Application of multidimensional process mining to internal logistics for a mixed-model assembly line.	Check and Discovery

2.6.2 Challenges of Mining Supply Chains

Mining business processes across the supply chain partners (cross-organisation) is a research challenge in the process mining research community [357]. According to Jain et al. [191], existing approaches cannot effectively design, operate and evaluate an agile supply chain due to the complex, stochastic, dynamic nature and multi-criteria of logistic processes involved within a global supply chain. There is a consensus that a global supply chain analysis is required. However, there has been little work on techniques for analysing business processes as a global supply chain [152]. It is therefore crucial that running an effective supply chain system requires supply chain process models [235] that account for the agility and holistic process in the supply chain network.

Another challenge is that there are several number of events that are generated at different granularity levels of a business process. Thus, it accounts for a high volume of complex data; hence, data processing and process analytics become a complex, and resource-intensive task [321]. This is because event data is stored in fragments in an information system. As a result, much effort will be needed to extract these pieces of event information, as data extraction is an integral part of any process mining efforts [357]. In addition, extracting process data from the event log, as a prerequisite step to process mining [211] is challenging. This is because, in various inter-organisational collaborations, it is not clear how supply chain event logs can be used across different participating companies [152].

2.6.3 Process Mining in the context of Supply Chain Optimisation

Discovering business processes across organisational boundaries has increasingly become an exciting business need. A detailed understanding of the business processes, in terms of frequencies and costs, can form a basis for supply chain optimisation [257]. However, describing such processes is non-trivial. As an effect, no party in the supply chain network has a good view of the whole set of activities executed at the different levels of the supply chain (e.g. warehousing, transportation planning and billing). Therefore, the overall supply chain business processes can be improved if the necessary information about the whole business process is analysed. Process mining is not limited to handling control flows, but other aspects like resources, data, organisational entities, decision points, and costs [357].

In obtaining an optimised holistic supply chain, individual parties can analyse process models with a deeper understanding of the entire distribution process. The analysis will enable the collaborating parties to effect changes at different levels of the supply chain in order to reduce the overall process costs [257]. For instance, supply chain logistics can significantly benefit from the application of process mining if the knowledge of the existing logistic processes can be used efficiently and effectively to improve competitiveness [25].

2.7 Optimisation Model-Based Methods - Metaheuristics

Optimisation model-based methods are mathematical models that search for the best solution from all feasible solutions. These methods have been applied in various contexts such as engineering, bioinformatics, operational research, geophysics, information retrieval, finance, economics, and management. Optimisation methods can be categorised into heuristics, metaheuristics and exact optimisation methods. In heuristics methods, algorithms are often effectively applied to a specific problem and ineffective to other problems. Metaheuristics, on the other hand, are generic optimisers which can be applied to almost all optimisation problems [2].

The word metaheuristics was introduced by Fred Glover [154] to illustrate heuristic methods with no problem-specific features. The idea of metaheuristics is to harmonise two search schemas: exploration (diversification) and exploitation (intensification) [36]. In terms of exploration, different regions of the search space is searched to create opportunities for diversification, whereas the term exploitation refers to the ability to obtain high quality solutions from known regions. The concept of exploration and exploitation is similar in machine learning algorithms which corresponds to the acquisition and utilisation of knowledge about unknown problems [376]. Despite these capabilities, there are varieties of optimisation problem types (i.e., continuous optimisation, discrete optimisation, dynamic optimisation, multi-objective optimisation, multitasking optimisation, and linked optimisation problems) that some metaheuristics cannot solve. This creates a need to develop variants of metaheuristics that adapt to such problem types.

Metaheuristics are algorithmic structure generally applied to various optimisation problems requiring only a few modifications to tackle a given problem. The abstraction of metaheuristics is presented in Algorithm 1, showing the trade-off between exploration and exploitation. An efficient search process is determined by how the trade-off between exploration and exploitation is properly managed, creating different metaheuristics classifications as shown in Figure 2.6. Abdel-Basset et al. [2] provides a new classification of metaheuristics into metaphor-based and non-metaphor based metaheuristics.

Algorithm 1: Algorithmic Framework for Metaheuristics [2]

```
initialisation (one or more solutions);
while Stopping criterion not met do
    if exploit then
        Create new solution by exploitation step
    else
        Create new solution by exploration step
    end
    Update best found solution
end
Result: return best from solutions
```

2.7.1 Metaphor Based Metaheuristics

Metaphor-based metaheuristics are algorithms that are driven by natural phenomena, and human behaviour in real modern life or mathematical concepts.

Biology Based Metaheuristics

Biological characteristics inspire most metaheuristics. Due to nature, biological agents have evolved, giving rise to intelligent behavioural and biological characteristics that allow them to undertake complex tasks, motivated by adaptability, self-learning, robustness and efficiency. Many researchers in the computational intelligence community have adopted different biological phenomena and processes to mimic such biological systems in tackling complex modelling, simulation and optimisation problems [105]. Abdel-Basset et al. in [2] stated three main paradigms that are exemplified by such biological traits. One is seen in the genetic inheritance process of organisms (evolutionary). The second arises from the social behaviour of animals such as ant colonies, beehives and bird flocks. They undertake self-organising tasks to explore complex search spaces to achieve a common goal (swarm). The third paradigm emphasises the human body's immune system or the brain's neural activity (immune systems). Detailed analysis is provided in Table 2.7.

Evolutionary Algorithms EAs are computational intelligence techniques which are generic population-based metaheuristics based on the principles of biological evolution. In EAs, no deep mathematical knowledge is required to tackle a problem, however no optimal solution

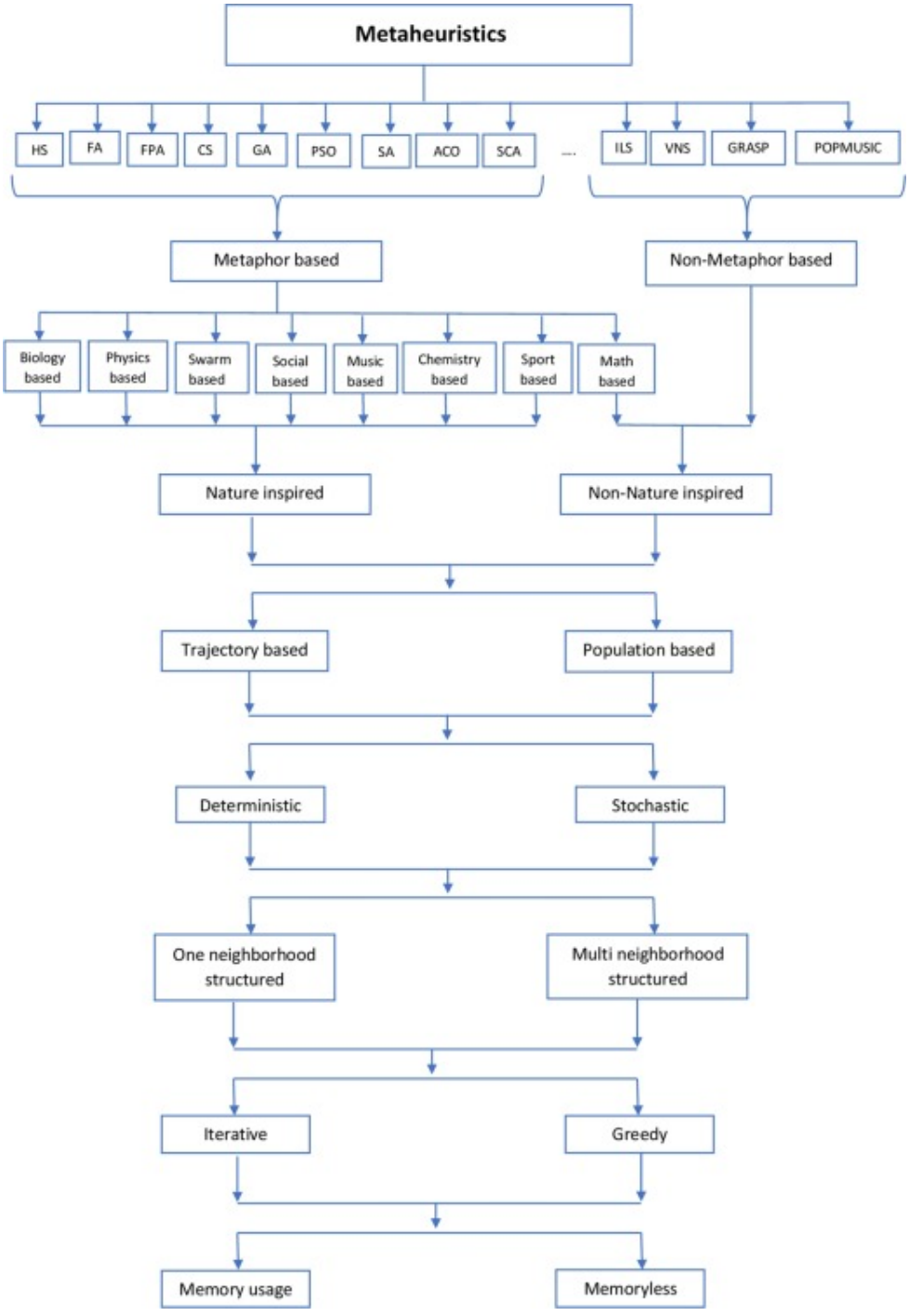


Figure 2.6: Meta-heuristics Taxonomies [2]

is guaranteed in finite time. EAs are suited for solving a broad range of complex problems characterised by discontinuity, non-linearity and multivariability [182]. A generic evolutionary algorithm is presented in Algorithm 2. The first step is population initialisation which involves random generation of solutions that constitutes the population. Next is the evaluation of individual solutions in the population using a fitness function. Each fitness

value explains the quality of the solution that was generated. Next, a loop iterates through generations which terminates whenever a stopping criterion is met. Inside the loop, individual parents are selected for breeding. The stronger the individual parent, the more chances the parent has to produce offspring for the next generation. Evolutionary operators are used in the next step to produce new offspring based on the genetic trait of the parents. The last part involves the selection of individuals for survival. There are variants of EAs, including genetic algorithm, evolutionary programming, genetic programming and evolutionary strategies. Details of these variants are shown in Table 2.7.

Algorithm 2: Generic Evolutionary Algorithm [182]

```
generation ← 0;
population initialisation;
population evaluation;
while Stopping criterion not met do
    generation ←generation+1;
    select individuals for reproduction;
    apply operators;
    evaluate new offspring;
    select individuals for survival;
end
```

Swarm Intelligence SI is inspired by the group behaviour of agents in a community, such as birds and insects. SI uses the principles of decentralisation where candidate solutions are updated through the local interactions with each other and the environment. The most common variants of SI are particle swarm optimisation PSO, ant colony optimisation ACO, artificial bee colony ABC algorithm, glowworm swarm algorithm GSA, firefly algorithm FFA, cuckoo search algorithm CSA, bat algorithm BA, and hunting search (HS) algorithm.

Artificial Immune System AIS is based on the theoretical framework of immunology and observed immune functions, principles and models. When adapted to optimisation, the antibodies represent the candidate solutions, which are iteratively evolved through repeating the operations of cloning, mutation and selection. The antigen represents an objective function, and a memory cell keeps the most promising solutions. There are variants of AIS based on the principle of clonal selection, including negative selection algorithm NSA, positive selection algorithm PSA, a clonal selection algorithm CSA, continuous immune network models, and discrete immune network models [2] [351].

Table 2.7: Summary of Biology-Based Metaheuristics

Paradigms	**Algorithms**	**Description**
Evolutionary Algorithms (EAs)	Genetic Algorithm GA [185] [368] [329] [4]	GA is a widely known EA initially introduced by John Holland [174]. GA is based on Darwin's principle, i.e., survival of the fittest. GA employs the process of biological evolution using the following operators; selection, crossover, and mutation. Chromosomes are candidate solutions for a given problem and are evaluated based on their fitness. The generation of new solutions is influenced by the parents selected for breeding. Several studies have shown the efficacy of GA in tackling supply chain optimisation problem[127].
	Evolutionary Programming EP [136]	EP focuses on the evolution of finite state machines. Fogel in [136] developed EP based on the principles of EAs, using the same evolutionary concepts.
	Genetic Programming GP [221]	The earliest application of GP created was used to tackle the problem of discovering finite-state machines. In genetic programming, the program can evolve by itself during the evaluation process. The search space is the space of computer programs, and the solution representation is a hierarchically tree-structured computer program. A solution is evaluated by running a program solution against a set of test cases. The fitness function is a sum of distances between the correct and produced results.
	Evolutionary Strategies ES [311]	ES was introduced and developed by Rechenberg [311]. The algorithm uses a single individual with real-valued vector encoding. Each of the real variables is changed by Gaussian mutation with a mean of zero and standard deviation σ.
Swarm Intelligence SI	Particle Swarm Optimisation PSO [130] [209]	PSO is introduced by Kennedy as social behaviour simulations and developed as an optimisation method in [209]. PSO algorithm starts by randomly searching and evaluating a particle population and keeps surviving for all generations until searching criteria are met [354]. Each candidate solution (particle) is explicitly associated with a search process, which has a velocity and a memory of the best positions it has so far.

	Ant Colony Optimisation ACO [89] [117]	ACO is used to tackle a problem with a combination character. ACO uses the concept of the nature of the ant colony, where each ant chooses the best value for every phase impacted by prior ants and the quality of each track [354].
	Other SI algorithms	ABC algorithm is inspired by the foraging behavior of honeybee colony [203]. Glowworm swarm optimization algorithm imitates the behavior of glowworms based on the changing intensity of luciferin emission [223]. Firefly algorithm mimics the idealised behavior of flashing characteristics of fireflies [142]. Cuckoo search algorithm is inspired by the reproduction strategy of cuckoo birds [380]. Bat algorithm is inspired by the echolocation behavior of bats based on pulse rates of emission and loudness [379]. Hunting search algorithm is inspired by group hunting predators like wolves [283].
Artificial Immune System	Clonal Selection Algorithm CSA [54]	The theory of CSA was first introduced in 1959 based on the adaptive immune system's basic response (lymphocytes) to antigenic stimulus. A population of antibodies (solutions) is randomly generated and evaluated. Next, the higher affinities antibodies are cloned to generate more antibodies against the antigen. The uncloned ones are replaced by new ones, and the best solutions are retained in a memory cell.
	Other algorithms	Different studies from the literature were brought together to provide a set of general-purpose AIS algorithms. These include; Negative Selection Algorithms, Optimisation version of Artificial Immune Network, a positive selection algorithm [330], continuous immune network models [129] [359], and discrete immune network models [101] [351].

Summary of other Metaphor Based Metaheuristics

Other metaphor-based metaheuristics are presented in Table 2.8, and they can be categorised into different groups, including chemistry, music, maths, physics, sport and social based.

Table 2.8: Summary of other Metaphor Based Metaheuristics

Paradigms	**Description**	**Variants**
Chemistry Based Metaheuristics	Chemistry based metaheuristic is motivated by the properties and behaviour of matter [225].	Chemical Reaction Optimisation (CRO) [225], Artificial Chemical Process ACP [187], Artificial Chemical Reaction Optimisation ACRO [5], Gases Brownian Motion Optimisation [1], Chemotherapy Science Algorithm CSA [327], Chemical Reaction Algorithm CRA [264]
Music-Based Metaheuristics	Music-based metaheuristics mimics the creative process of musical innovation based on music rules, processes, concepts, and events.	Harmony Search HS [148], Melody Search MS [12], Method of Musical Composition MMC [276]
Math-Based Metaheuristics	Maths-based metaheuristics are optimisation techniques made by interoperation of metaheuristics and mathematical programming techniques.	Matheuristics [43], Base Optimisation Algorithm BOA [326], Sine Cosine Algorithm SCA [271], Simulated Kalman Filter Algorithm SKFA [181], Golden Sine Algorithm GSA [349]
Physics-Based Metaheuristics	Physics-based metaheuristics adopt the principles and theories of physics and apply them to solve real-world optimisation problems.	Simulated Annealing SA [214], Stochastic Diffusion Search SDS [35], Self-Propelled Particles [362], Extremal Optimisation EO [38], Intelligent Water Drops IWD [177], Central Force Optimisation CFO [138], River Formation Dynamics RFD [302], Gravitational Search Algorithm GSA [309], Charged System Search CSS [205], other variants can be found in [58].
Sport-Based Metaheuristics	Sport-based metaheuristics are novel, efficient search and optimisation techniques which are inspired from various sports processes, rules, events, and concepts.	Soccer League Competition Algorithm SLCA [275], League Championship Algorithm LCA [204], Golden Ball GB [286], Football Game Algorithm FGA [126], Tug of War Optimisation TWO [206]
Social Based Metaheuristics	Social-based metaheuristics stimulate human behaviour based on how users of social networks interact.	Cultural Algorithm CA [314], Grammatical Evolution GE [323] , Imperialist Competitive Algorithm ICA [15], Social Emotional Optimisation Algorithm SEOA [377], Fireworks Optimisation Algorithm FOA [347], Artificial Tribe Algorithm ATA [74], Teaching–Learning-Based Optimisation TLBO [307]

2.7.2 Non-Metaphor Based Metaheuristics

Non-metaphor metaheuristics do not rely on any inspration from nature or physical characteristics to determine their search strategy. Several techniques have been discussed in the literature where algorithms require no stimulation for any search process. Examples include; Tabu search TS [155], Scatter search SS [153], Guided Local Search [366], Greedy Randomised Adaptive Search GRAS [133], Iterated Local Search ILS [245], Variable Neighborhood Search VNS [273], Cross-Entropy Method CEM [322], Partial Optimisation Metaheuristic under Special Intensification Conditions POMSIC [344], etc. Table 2.9 discusses three of these algorithms which have been applied extensively to real-world problems.

Table 2.9: Examples of Non-Metaphor Metaheuristics

Algorithms	**Description**
Tabu Search TS [155]	Glove and McMillan introduced TS [155]. The algorithm is a non-nature-inspired metaheuristic based on the idea of prohibition (tabu or taboo) of already visited search areas from being revisited to promote diversification. TS searches the neighbourhood of the current solution to get a new one with an improved functional value and maintains a tabu list of solutions obtained in previous iterations.
Iterated Local Search ILS [245]	ILS iteratively builds a sequence of solutions generated by an embedded heuristic (local search), leading to a far better solution. ILS is based on a local search strategy using a single solution along the iterative process. ILS starts with random solution generation within a predefined search space, and the number of iterations defines the stopping criterion. The concept of ILS is defined by the steps of local search, perturbation, and the stopping criterion.
Guilded Local Search GLS [366]	GLS adopts a penalty-based approach to interact with the improvement procedure. The interaction allows a process of escape from the local optima, which improves the efficiency and robustness of the underlying local search algorithm.

Despite several successful applications of metaheuristics (metaphor-based and non-metaphor-based) in all spectrums of complex problems, the majority of the research studies have not considered the complexity that is associated with real-world supply chain problems in terms of interacting decision components. The obvious reason is that the algorithms proposed in the literature do not incorporate the notion of a network of interconnected optimisation problems seen in supply chains. Similarly, no complex computational analysis has provided insights into the interactions between different optimisation problems [41]. Such interacting decision-making systems need to be considered in an optimisation framework. Therefore, a formal conceptual framework for interconnected optimisation problems

is inevitable. In other words, the theoretical investigation of computational methods for such linked optimisation problems poses a new challenge, requiring an appropriate framework.

2.7.3 Multi-Objective Optimisation Problem MOOP

In the last 30 years, there has been growing research interest in studying problems comprising several conflicting objectives [105]. The multi-objective optimisation problem simultaneously considers optimisation problems with two or more objective functions. MOOP is generally applied to decisions with two or more simultaneous objectives [281]. The application area includes economics, logistics, and many engineering and science problems. Multi-objective problems involve a decision-making process of selecting a possible course of action to obtain multiple objectives or goals while satisfying the constraints dictated by the environment. Multi-objective problem structure is represented in Figure 2.7.

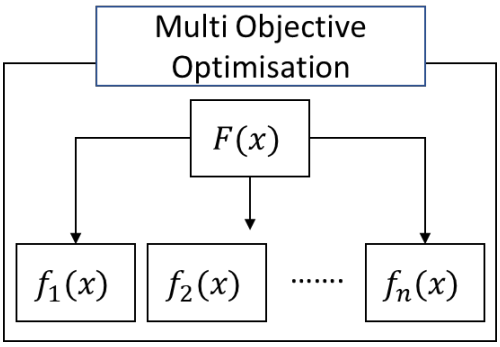


Figure 2.7: Multi-Objective Optimisation Problem

In a supply chain, decisions usually involve multiple and conflicting objectives such as cost, service level, and resource utilisation. Different strategic designs are adopted for the supply chain multi-objective problems. For example, Bandyopadhyay and Bhattacharya [22] use possibilistic non-linear programming (PNLP) to solve the minimisation of the total cost and bullwhip effect of a two-echelon supply chain model. Also, Zhang and Xu [400] investigated the quantity discount policy in the management of supply chain under complex fuzzy environment and address the problem a possibilistic or fuzzy non-linear programming. Chen [72] developed a fuzzy mixed-integer non-linear programming (FMINP) model to tackle a multi-echelon supply chain network problem driven by uncertain market demands and product prices. Mirzapour et al. [272] addressed a production-distribution planning problem under uncertainty as a stochastic mixed non-linear programming (SMNLP).

Different authors have classified multi-objective optimisation methods according to different criteria [267]. Cohon [88] classified multi-objective optimisation methods into two distinct subsets: generating methods and preference-based methods. Rosenthal [318] identifies three classes of multi-objective opimisation methods: partial generation of the Pareto optimisation set, explicit value function maximisation and interactive implicit value func-

tion maximisation. Hwang and Masud [180] provides an extensive classification of methodologies used in multi-objective optimisation and group them into four classes - no preference, a posteriori, a priori, and interactive methods. The no preference methods use some relatively simple method without considering the opinions of the decision maker when solving the multi-objective optimisation problem [267]. These methods are suitable whenever there are no special expectations required by the decision maker. Examples of methods presented in this category are global criterion [385], and multi-objective proximal bundle method [268]. The a posteriori methods are Pareto-based methods that use a selection of mechanism for Pareto-optimality. Here the decision maker is presented with Pareto optimal set to select preferred solution among the alternatives [267]. Some methods proposed in this class include: weighting [146], ε-constraint [162], hybrid (combination of weighting and ε-constraint) [95], hyperplane [383] and scalarisation [291] methods. In a priori methods, decision maker's preferences, hopes and opinions are considered before the solution process. Example of methods in this class include: value function [207], lexicographic ordering [135], and goal programming [96] methods. The interactive methods consider the interactive process of information processing involving the decision maker and an analyst or interactive computer program.

Most of the multi-objective methods identified in the literature consider single optimisation problems with many objectives because there are no explicit modelling of the interactions between decision-making systems in interconnected optimisation problems [300]. So, the modelling of such interactions is an opportunity to provide computational extensions to current optimisation paradigms. To the best of our knowledge, there are different approaches to applying an algorithm to solving such interacting decision-making systems. One of the approaches may require the combination of the corresponding solutions of the interacting problems as a joint or holistic solution where the joint solution is evaluated using a holistic fitness function. The challenge in this approach is concerned with identifying the best way to combine them, considering that solutions have completely different solution spaces. Hence, the problem representation is critical; therefore, more investigation would be needed. In addition, the designation of operators for the specific solution representation also poses a potential problem. A typical way this can be viewed is to consider the problem as multi-objective optimisation problem. This requires re-modelling the problem as a joint or a single solution-based problem containing several optimisation goals. However, the presence of multiple solution spaces introduces several interesting complexities in designing a search algorithm for such problems [102].

2.8 Multi-criteria Decision-Making Methods

A decision maker explores a set of alternatives to determine an optimal decision based on some criteria. Multi-criteria Decision-Making MCDM methods are developed to standardise a complex decision-making process based on the alternative evaluation theory. MCDM

methods provide an explicit, efficient and rational process of decision-making so as to improve the quality of a decision [194]. Many MCDM methods in the literature are used in application areas like supply chain decision-making. Most commonly used methods as reviewed in [65] include; Preference Ranking Organisation Method for Enrichment Evaluation (PROMETHEE) [46], Elimination Et Choix Traduisant la Realité (ELECTRE)[320], Weighted Sum Method (WSM) and Weighted Product Method (WPM), Analytic Hierarchy Process (AHP) [324], Full Multiplicative Form Multi-Objective Optimisation by Ratio Analysis (MULTIMOORA) [47], Technique for Order Preference by Similarity to Ideal Solutions (TOPSIS)[66], VIKOR [285]. The adoption of these methods in different supply chain decision-making levels includes; supplier selection, manufacturing, warehousing, and logistics [212].

2.9 Summary

In summary, this chapter provides some concepts of supply chain management and present supply chain decision-making problems addressed in the literature. Specifically, this chapter uses the supply chain decision problems to describe the concept of a linked optimisation problem. The concept is used to explain the issues relating to interacting decision-making systems. Three issues were reviewed relating to linked problems in real world. The first review considers several techniques for linking supply chain fragmented data proposed in the literature to identify data relationships. Fragmented data occurs due to business interactions between supply chain participants. The interactions involve several processes, resulting in hundreds or thousands of tables and attributes in a database. The next issue entails understanding the process sequence of the supply chain. This is believed that partial information about the business processes is captured by the information system of the individual supply chain collaborators. Therefore, each partner does not have an insight into the overall processes. To understand the overall business process of the supply chain, this chapter reviews several techniques proposed in the literature that mine and learn the entire processes of the supply chain. The mining process could use a common reference point for each supply chain partner and combine the partial information captured by their information systems to infer an overall distributed process. Finally, several state-of-the-art algorithms were explored in the literature, and these algorithms have been used to tackle problems relating to supply chain problems. The research investigates the major state-of-the-art optimisation algorithms presented in the literature. The understanding of these algorithms is to design suitable algorithmic approaches inspired by the existing algorithms and the nature of problem linkages to address different problem linkages in supply chains.

3

Supply Chain Optimisation: A Linked Problem Perspective

3.1 Introduction

Increasingly, business decisions are being automated or strongly supported by optimisation algorithms. However, most of these algorithms do not incorporate the complexity associated with interacting decision-making systems. It is well-known that decisions made at one point in supply chains can have significant consequences that ripple through linked production and transportation systems. Recently, global shocks to supply chains (e.g., COVID-19, climate change, blockage of the Suez canal) have demonstrated the importance of these interdependencies and the need to create supply chains that are more resilient and have significantly reduced impact on the environment. There is a growing literature spanning several research communities that studies together two or more optimisation problems whose solutions interact in some way. This chapter develops a general formalism for linked optimisation problems. It allows systems of linked problems to be defined with a precise specification of linkages in terms of solution domain, objective functions and constraints.

The concept of linked decision-making arises in several different research communities. In economic theory, for instance, the theory of exchange networks conceptualises multiple actors networking through exchanges to achieve valuable outcomes. Cook and Whitmeyer's [92] study of exchange theory and network analysis, characterised an exchange network by interaction, structure and order. Similarly, the study of network exchange models in [53] explains how networks of individuals exchange resources to improve prior conditions. Cook and Whitmeyer further observed that this exchange network focuses on the ties between members. In other words, according to Zelbst et al's [390] study of supply chain linkages, these ties consist of a set of interactions linking exchange relations, via the distribution of valued resources among different actors (persons or corporate groups), across a single network. As stated in [140], exchange networks are measured and understood in terms of the linkages in the network [251].

In the complex systems community, a central focus of the study is systems containing

large parts (subsystems) that interact in a non-trivial way [337] with emergent properties arising from the network of interactions. This system could be attributed to one or more features: (1) "significant interactions", (2) "high number of parts or interactions", (3) "nonlinearity", (4) "broken symmetry", (5) "nonholonomic constraints" [45, p. 79]. In [45, 339], a system is often viewed in terms of its complexity, i.e., detail (combinatorial) complexity and dynamic complexity corresponding to two or more optimisation problems. In detail complexity, the system usually contains several unique components. In contrast, dynamic complexity involves uncertainty in the system's response to a set of inputs arising from the interactions among the system components over time. From this point of view, a linked problem could be characterised as a complex system if it contains several interacting unique components that manifest uncertainty.

From the operational research perspective, a linked problem is typically a decision problem made up of interconnected components [341]. Recently, within the evolutionary computation research community, the linked problem is being identified with increasing frequency, so it is important to recognise what the linked problem is, what its key research challenges are and improve our understanding of how to solve it. Therefore, this chapter aims to recognise and provide a definition of linked optimisation problems from a supply chain viewpoint.

From a common point of centrality, researchers from the above-mentioned research domains have extended the concept of linkages to service and supply chain because, by definition, according to [390], a supply chain is an interrelated network of suppliers and customers plus other factors such as logistics etc. As our prime example, supply chain exhibits linkages among members of a supply chain network. As observed in [390], these linkages are created to fulfil a requirement for some resources.

A supply chain is a broad term that explains a business ecosystem's various functions (e.g., warehousing, transportation planning and billing). This term forms a significant part of business operations. A company's success and customer satisfaction depend on how well supply chain activities are managed. Thus, as stated in Lau and Song's [228] study of supply chain optimisation, supply chain requires guided coordination in order to exploit economies of scale and other benefits. However, in the study of evolutionary algorithms for supply chain optimisation in [182], it was noted that in managing a supply chain, so many individual decisions have to be regularly made at intervals, which are different in scope and importance. Therefore, according to [355], finding an optimal strategy that addresses the whole supply chain problem becomes more challenging as decisions are linked. Specifically, managing the activities at the various levels of the chain of companies is challenging. This is because there are different sources of uncertainty, and bullwhip effect exists between the various entities of the supply chain [23, 355]. A bullwhip effect involves increased orders' variability as orders move upstream in the supply chain [371].

We utilise a supply chain network to broaden our understanding of a linked problem and the challenges it raises. For instance, Choi et al [79] conceptualises a supply chain as

a complex adaptive system. They argue that for a supply chain network to be considered a complex adaptive system, there must be evidence of interplay between an internal mechanism (system) and its environment and the co-existence between the system and the environment. According to them, it means that a supply chain, as a complex adaptive system, must be emerging, self-organising, dynamic, and evolving. However, based on the authors' claims, the entire supply chain network has been challenging to manage, and this is largely caused by a lack of complete understanding of the supply chain network. In addition, a supply chain is characterised by the presence of detail complexity and dynamic complexity, which is driven by the consequence of the interactions between supply chain components. As noted by [45], these interactions form the linkages, which are the sources of the dynamic complexity. Focusing this research on the linkage characteristics of the supply chain forms one of the crucial motives for this study.

Similarly, the process mining research community recognised that linkages exist in a supply chain. For example, the study of workflow process mining in [213] noted that components in a supply chain network are closely related such that a slight change in one component can significantly affect the logistic operations of another. In other words, as claimed in [227], attributes in one aspect of the supply chain network interact with elements in the other levels of the supply chain. Hence, if one pays too much attention to one aspect of the supply chain, obtaining an effective process formulation for the entire supply chain network may fail. According to [191], existing approaches cannot effectively design, operate and evaluate an agile supply chain due to the complex, stochastic, dynamic nature and multi-criteria of logistic processes involved within a supply chain network. One of the reasons for this is that the overall chain of business processes, such as scheduling, ordering, delivering, and allocation, is widely distributed and executed by several parties across all the supply chain participants. Therefore, no individual collaborator has a complete overview of the supply chain logistics or processes [257]. There is, therefore, a consensus that a complete supply chain analysis is required. However, there has been little work on techniques for analysing business processes as a global supply chain [152].

Increasingly, decisions are automated or strongly supported by optimisation algorithms - manufacturing optimisation, b2b ordering, financial trading, transportation scheduling and allocation. However, most of these algorithms do not incorporate the complexity associated with interacting decision-making systems like supply chains [184]. Such interacting decision-making systems need to be considered, and we do so through optimisation. The reason for this is that the interactions between such decision-making systems are not modelled [300]. So, we believe that modelling such interactions is an opportunity to provide computational extensions to current optimisation paradigms. On this note, the dynamic nature of supply chain components and interactions is a strong prospect for tackling such problems. To the best of our knowledge, no study has yet provided an explicit unified model that defines the features of linked problems and how such components affect each other. So, the abstraction of the linkages in a supply chain is worth addressing.

This chapter makes the following contributions; first, we introduce the nomenclature "Linked Optimisation Problem" to identify optimisation problem sets that interact to provide a general concept that unifies research on interconnected optimisation components. We then define a linked problem and introduce adjacency matrices to characterise the interactions between the sub-problems with real-world examples. Finally, we introduce the notion of holistic goals for linked optimisation and postulate an external regulator to optimise the holistic goal. The rest of the chapter is structured as follows. Section 3.2 provides a definition of a linked optimisation problem with real-world examples. In Section 3.3, we present existing algorithmic approaches and techniques for linked optimisation problems. In Section 3.4.1, we present a holistic perspective of linked optimisation in terms of external regulators. Section 3.5 presents some key research questions and directions. Section 3.6 gives a summary of the chapter.

3.2 Linked Optimisation Problem

Several research communities - evolutionary computation [300], operational research [68], information sciences [382], transportation research [27], information systems [167] - have formulated and aimed at solving linked problems. Each have used different terms to reference this type of problem such as “global optimisation” [183], "large scale optimisation problem" [382], “joint optimisation” [312], “coupled optimisation problem” [27], “coordination” [68, 125], “combined problem” [242, 169], “multi-silo optimisation” [182], “multi-stage optimisation” [292], “cooperative subcomponents” [300], “integrated optimisation problem” [167, 389], “bilevel problem” [346], "multi-component optimisation" [363], "multitasking optimisation" [284] and "multi-echelon" [6]. We believe this makes it difficult to create a structured global research effort on the topic. This is why we propose to use the more general term "Linked optimisation".

By providing an abstracted, general concept that unifies research on interconnected optimisation components/units as defined in Section 3.2.1, this chapter provides the research community with the term "Linked Optimisation Problem". This will consequently improve research quality in this area.

3.2.1 Linked Problem Definition

A supply chain exhibits linkages between the individual operating components. Thus, an optimal solution for a given component may not guarantee an optimal solution for the overall problem [363]. According to Ackoff in [3], "problems are abstracted from systems of problems, messes. Messes require holistic treatment. They cannot be treated effectively by decomposing them analytically into different problems to which optimal solutions are sought". Therefore, solving a linked problem requires a holistic approach. As shown in Figure 3.1, the interaction among the components of a linked optimisation problem is achieved

by the level of influence a solution of one problem has on the other problems in the linked structure. There are three possibilities; one is change induced on the structure of the problem solution, another is the change induced on the objective function, or/and the imposition/changes to problem constraints.

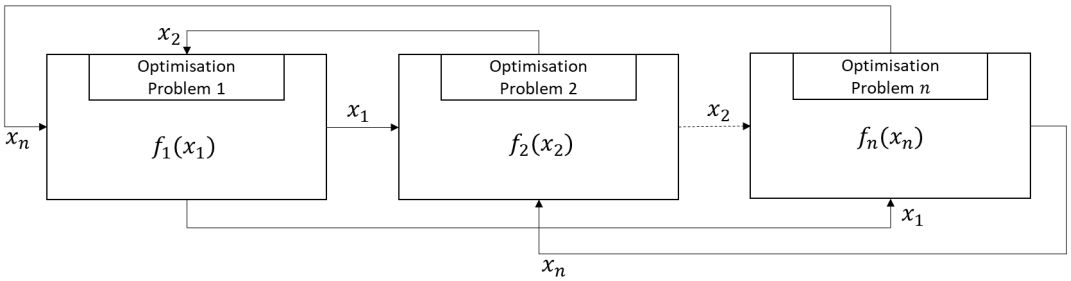


Figure 3.1: Illustration of the Linked Optimisation Problem

In general, we denote each optimisation problem in Figure 3.1 as p, defined by the tuple:

$$p = \{x, f, c\} \tag{3.1}$$

where

- x - is a search space of p, i.e. the set of possible solutions to the problem.
- $f: x \rightarrow \mathbb{R}$, an objective function which evaluates solutions in x by mapping each solution to a real number.
- c - a set of constraints that determines the boundaries where feasible solutions lie within the search space. A solution is feasible if it satisfies the constraint set c.

Let's assume now a set of n (for any $n \geq 2$) problems $p_1, ..., p_n$ which are somehow linked based on the linked structure in Figure 3.1. We define a link between two problems when the decision made for one problem p_i, i.e. the chosen solution x^i_*, affects another problem p_j. We identify that the decision made in p_i can affect either or various elements of p_j, its search space x^j, its evaluation function f^j, and/or its constraints c^j. Equation 3.1 can be rewritten by:

$$p_i = \left\{ x^i_{\{x^1_*, \cdots, x^n_*\} \setminus x^i_*}, f^i_{\{x^1_*, \cdots, x^n_*\} \setminus x^i_*}, c^i_{\{x^1_*, \cdots, x^n_*\} \setminus x^i_*} \right\} \tag{3.2}$$

where:

- x^i_* represents any candidate solution in x^i.
- $\{x^1_*, \cdots, x^n_*\} \setminus x^i_*$ are individual feasible solutions from other problems which changes the behaviour of p_i. The impact of the solutions on p_i may **affect/modify** p_i's search space x^i, objective function f^i or constraints c^i. If p_i is an independent problem then, $\{x^1_*, \cdots, x^n_*\} \setminus x^i_* = \emptyset$.

- $x^i_{\{x^1_*,\cdots,x^n_*\}\setminus x^i_*}$ is a search space of problem p_i. x^i may change with respect to solutions $\{x^1_*,\cdots,x^n_*\}\setminus x^i_*$, provided that, p_i is a dependent problem. The changes in x^i can either be explicit or implicit. It is explicit if changes alter the representation of the solutions (i.e., changes in dimensionality) and implicit if changes simply modify the location of an optimum without altering the search space representation.

- $f^i_{\{x^1_*,\cdots,x^n_*\}\setminus x^i_*} : x^i \rightarrow \mathbb{R}$ represents the objective function. In certain situations, the objective function may change with respect to $\{x^1_*,\cdots,x^n_*\}\setminus x^i_*$, provided that, $\{x^1_*,\cdots,x^n_*\}\setminus x^i_* \neq \emptyset$. A change in the objective function is evident if certain changes in the landscapes occur whenever a solution from another problem influences it.

- $c^i_{\{x^1_*,\cdots,x^n_*\}\setminus x^i_*}$ represents the constraints set of p_i. In some cases, a solution to one problem may affect one or more constraints of another problem. In our definition, a change in constraints c^i may be attributed to the impact of $(x^*_1,\cdots,x^*_n)\setminus x^i$, which causes some feasible regions around the optimum to become infeasible. In other words, a solution to a problem can change the feasible state of solutions to another problem.

We represent a linked optimisation problem as P, containing n related problems. The connectedness of the problems is described by $D = \{D^X, D^F, D^C\}$, so that P can be defined as:

$$P = \{p_1, p_2, \cdots, p_n, (D)\} \tag{3.3}$$

D is a set of 3 adjacency matrix representations of the linkages between problems in P. The representation of each matrix in D describes what feature of the problem is changed or altered. The first matrix below, D^X, represents an $n \times n$ adjacency matrix. The matrix provides information on how a solution influences the solution to another problem. Let $D^X_{ij} \in \{0,1\}$, such that, D^X_{ij} takes the value of 1 if a solution x^i_* of problem p_i affects the solution set x^j in problem p_j and 0 otherwise.

$$D^X = \begin{Bmatrix} D^X_{11} & D^X_{12} & \cdots & D^X_{1n} \\ D^X_{21} & D^X_{22} & \cdots & D^X_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ D^X_{n1} & D^X_{n2} & \cdots & D^X_{nn} \end{Bmatrix}, D^F = \begin{Bmatrix} D^F_{11} & D^F_{12} & \cdots & D^F_{1n} \\ D^F_{21} & D^F_{22} & \cdots & D^F_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ D^F_{n1} & D^F_{n2} & \cdots & D^F_{nn} \end{Bmatrix}, D^C = \begin{Bmatrix} D^C_{11} & D^C_{12} & \cdots & D^C_{1n} \\ D^C_{21} & D^C_{22} & \cdots & D^C_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ D^C_{n1} & D^C_{n2} & \cdots & D^C_{nn} \end{Bmatrix}$$

Similarly, D^F represents an $n \times n$ adjacency matrix of connected problems in P where the choice of a solution for one problem changes the value of the objective function of another problem. We denote D^F_{ij} as a variable that takes the value of 1 if solution x^i_* changes the value of the objective function f^j in problem p_j and 0 otherwise.

The third adjacency matrix is an $n \times n$ matrix D^C which specifies the constraints that change due to the choices of solutions from other problems. D^C_{ij} takes 1 when a solution x^i_* changes one or more constraints in c^i of another problem, and D^C_{ij} takes 0 when x^i_* has no impact on the behaviour of c^j.

$$[h] D^X_{ij} / D^F_{ij} / D^C_{ij} = \begin{cases} 1, & \text{if } x^i_* \; changes \; x^j / f^j / c^j \\ 0, & Otherwise \end{cases}$$

It is important to note that the definitions we considered have looked into n static problems in a linked optimisation problem. However, a linked optimisation problem could have a mix of static and time-dependent problems or can contain a collection of time-dependent problems.

An optimisation problem can either be static or dynamic. In a static optimisation problem, the problem's specifications do not change over time. However, in a dynamic optimisation problem, the features of the problem are time-variant; that is, the problem changes over time. The underlying dynamic characteristics of such a problem can be expressed differently. The characteristics include; time-linkage (i.e., current or past solutions determine the behaviour of a problem) [282], time-varying solution space (e.g., changes in dimensionality), time-varying objective function and time-varying constraints. These attributes are related to linked optimisation problems.

3.2.2 Real-World Examples of Linked Problems in Supply Chain

We will focus on five major supply chain problem types: Inventory problems (Production and Distribution problems), Container terminal problems (Truck scheduling, Routing and Storage allocation problems), Warehouse problems (Batching, Routing, Storage allocation and sequencing problems), Workforce and Service Systems (Scheduling, Routing, Allocation and Capacity Planning), and Offshore problems (Ship routing, Scheduling and Cargo allocation problems). These five supply chain problem types are considered based on their real-world importance at academic and industry levels. Some types have similar individual optimisation problems, while others differ in their functionalities or operations in the supply chain types.

Combining Lot Sizing and Vehicle Routing Problems [228]

This problem combines two classical NP-hard optimisation problems: Dynamic Lot sizing and Vehicle Routing problems. The essence is to identify how the total cost in the combined system is minimised over the planning horizon.

Problem one - Lot Sizing (p_1). This is an inventory allocation problem. The solution determines lot sizes to meet demands over a period of time. The aim is to determine the lots to balance replenishment with holding costs at the beginning of each period. Problem one is subject to a set of constraints, including; demands per period and inventory capacity

of each retailer. The solution of the lot sizing problem determines an instance of the routing problem p_2.

Problem two - Routing (p_2). This is a pickup and delivery problem with time windows. An instance of the routing problem p_2 is determined by the solution of the lot sizing problem p_1. The solution determines how vehicles are routed to deliver the lots to the retailers. The aim is to minimise routing costs. It is subject to a set of constraints, including; vehicle capacity and pickup and delivery time windows.

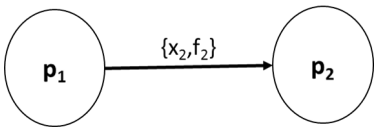


Figure 3.2: Lot Sizing and Vehicle Routing Linkages

Figure 3.2 shows the link between the two problems. The problems are linked so that the set of lots allocated to individual retailers forms the basis for generating an instance of the routing problem in delivering the lots to the retailers. The two adjacency matrices presented below show the features of problem two (p_2) affected by the link structure.

$$D^X = \begin{Bmatrix} 0 & 1 \\ 0 & 0 \end{Bmatrix}, D^F = \begin{Bmatrix} 0 & 1 \\ 0 & 0 \end{Bmatrix}$$

Integrated Machine Scheduling and Vehicle Routing Problem [356]

This problem is an integration of machine scheduling and vehicle routing problems which seek to minimise the total tardiness in completing the delivery of a set of jobs. Individual problems have well been studied with several variations and characteristics.

Problem one - Machine Scheduling (p_1). Machine scheduling is a classical job scheduling optimisation problem. The solution determines the scheduling of jobs to a given set of machines. The aim is to minimise the total completion time. The problem is subject to constraints associated with jobs and machines. The solution to the machine scheduling problem constrains an instance of the routing problem p_2.

Problem two - Routing (p_2). Routing is a vehicle routing problem with a time window. An instance of the routing problem is constrained by the solution of the machine scheduling problem. The solution determines the best tour to deliver a set of jobs to customers. The aim is to minimise total tardiness. Problem two is subject to constraints relating to the vehicles and tours.

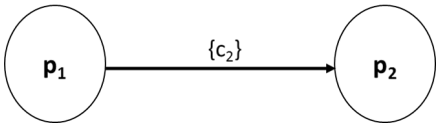


Figure 3.3: Machine Scheduling and Vehicle Routing Linkages

The link structure of the two problems is shown in Figure 3.3. The adjacency matrix representation D^C shows how a solution to problem one constrains an instance of problem two.

$$D^C = \begin{Bmatrix} 0 & 1 \\ 0 & 0 \end{Bmatrix}$$

Workforce Scheduling and Routing Problems [375]

This problem involves the integration of workforce scheduling and routing problems. The integration aims to minimise the operational cost comprising routing and outsourcing costs.

Problem one - Workforce Scheduling (p_1). This problem is a classical job assignment problem. The solution determines the scheduling of technicians to a set of tasks. The aim is to minimise the completion time of all tasks. It is subject to a set of constraints like skills requirements and tasks time windows. The solution to the workforce scheduling problem constrains an instance of the routing problem p_2.

Problem two - Routing (p_2). p_2 is a vehicle routing problem with time windows. A routing problem instance is constrained by the solution of the workforce scheduling problem. The solution of p_2 determines the best routes travelled by technicians to perform tasks. The aim is to minimise travelling costs. This is subject to constraints relating to route duration and time window.

Figure 3.4 shows the relationship structure between the two problems, and the adjacency matrix D^F depicts the level of interactions between the two problems. The feasibility of solutions in the routing problem p_2 is determined by the solution of the workforce scheduling problem.

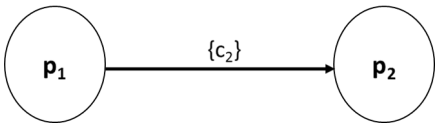


Figure 3.4: Workforce Scheduling and Routing Linkages

$$D^C = \begin{Bmatrix} 0 & 1 \\ 0 & 0 \end{Bmatrix}$$

Crane Handling Scheduling and Truck Routing Problem [73]

This problem involves the interrelationship between crane handling scheduling and truck routing problems. The problem as a whole seeks to find a solution that minimises the makespan for the loading and unloading a set of ships in a given time horizon.

Problem one - Crane Handling Scheduling (p_1). This is a typical scheduling problem. The solution determines the sequence of cranes to service a given set of containers. It is aimed at minimising the makespan of the set of containers. The problem is subject to a set

of constraints relating to container handling times. The solution of the crane scheduling determines an instance of the truck routing problem p_2.

Problem two - Truck Routing (p_2). Truck routing is a classical vehicle routing problem. The crane handling scheduling problem determines the problem instance of the truck routing. The solution determines the truck routes and assigns yard trucks to a set of containers. The aim is to minimise the travel time of the yard trucks. The solution of the yard truck routing problem affects the fitness value of the solution of the crane scheduling problem.

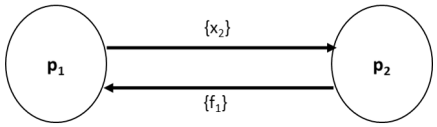


Figure 3.5: Crane Scheduling and Truck Routing Linkages

The scheduling solution of the cranes provides additional information about the precedence relations of the transportation task for each container. As depicted in figure 3.5, the information is used in p_2 to assign trucks to the containers and determine the transportation routes of the yard trucks. The two adjacency matrices below provide the level of interaction between the two problems on how they both influence each other.

$$D^X = \begin{Bmatrix} 0 & 1 \\ 0 & 0 \end{Bmatrix}, D^F = \begin{Bmatrix} 0 & 0 \\ 1 & 0 \end{Bmatrix}$$

Integrated Order Batching, Sequencing and Routing Problem in Warehouse Operations [75]

This problem is a picking operation optimisation in warehouses that involves the integration of order batching, batch sequencing and picker routing problems. The problem seeks to minimise the total tardiness of customer orders.

Problem one - Order Batching (p_1). The problem is a typical batching problem. The solution combines a set of customer orders into several batches. The problem aims at assigning customer orders to each batch based on predetermined batch size. This problem is subject to constraints, including order storage location and vehicle capacity. The solution of the batching problem determines an instance of the sequencing problem p_2.

Problem two - Batch Sequencing (p_2). p_2 is a classical sequencing problem. The problem instance of the sequencing problem p_2 is determined by the solution of the batching problem p_1. The solution to the batch sequencing problem determines the picking sequence of a given set of batches. The aim is to determine the shortest completion time of all batches. This is subject to a set of constraints relating to orders, due dates and batch-related restrictions. The solution of the batch sequencing problem determines an instance of the routing problem p_3.

Problem three - Picker Routing (p_3). The problem is a typical travelling salesman problem. The instance of the routing problem for each batch is determined by the solution of the sequencing problem p_2. The solution to the picker routing problem determines the picking sequence for each batch. The aim is to minimise the total travelling time for all batches. The problem is subject to a set of constraints relating to tours.

Figure 3.6 shows the structure of linkages between the three problems. The level of interactions between the problems is represented in the two matrices below.

$$D^X = \begin{Bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{Bmatrix}, D^F = \begin{Bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{Bmatrix}$$

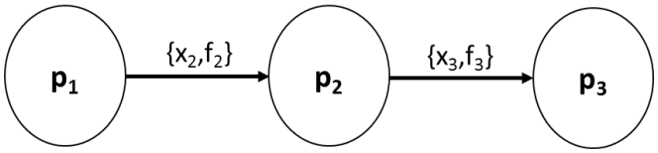


Figure 3.6: Order Batching, Batch Sequencing and Picker Routing Linkages

Solving Location, Allocation, and Capacity Planning Problems in Service Systems [240]

The problem involves three decision levels; several facility locations, demand allocation, and capacity requirement. The problem seeks to determine the minimum total capacity for a different number of facilities to fulfil a given level of service.

Problem one - Facility Location (p_1). The solution determines the number of facilities and locations. The aim is to minimise the number of facilities to service a given set of demand units. p_1 is subject to a set of constraints relating to demand units. The solution to the facility location problem constrains an instance of the demand allocation problem p_2 and the capacity requirement problem p_3.

Problem two - Demand Allocation (p_2). This problem is an assignment problem. The solution of the facility location constrains an instance of the demand allocation. The solution allocates demand sites to a defined number of facilities. p_2 seeks to minimise the total average travel time directly from the facility. This is subject to a set of constraints relating to the demand units. The solution to the demand allocation problem determines an instance of the capacity requirement problem p_3.

Problem three - Capacity Requirement (p_3). The problem instance is determined by the solution to problem p_2 and constrained by the solution of the facility allocation problem p_1. The solution finds individual facilities' capacity levels (number of service units). The aim is to minimise the total capacity required in each facility. p_3 is subject to a set of constraints relating to the facilities and the demand units.

The linked structure of the problem is shown in figure 3.7, and we provide below three adjacency matrices to represent the level of interactions between the three problems.

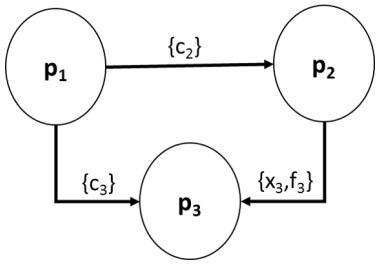


Figure 3.7: Location, Allocation and Capacity Problem Linkages

$$D^X = \begin{Bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{Bmatrix}, D^F = \begin{Bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{Bmatrix}, D^C = \begin{Bmatrix} 0 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{Bmatrix}$$

Freight Assignment and Routing in Tramp Shipping [242]

This problem combines two classical optimisation problems; assignment problem and routing problem relating to offshore supply chains. The problem seeks to maximise total profit.

Problem one - Freight Assignment (p_1) is an assignment optimisation problem. The solution determines the cargo assignment to ships seeking to maximise total cargo revenue. The assignment of cargoes to ships is subject to the ship's attributes in terms of capacity and configuration. The solution to the assignment problem forms an instance of the ship routing problem p_2

Problem two - Ship Routing (p_2). This is a routing problem with pickup/delivery (P/D) time windows. The instance of the ship routing problem is determined by the solution of the freight assignment problem p_1. The routing problem seeks to find the best routes to reduce shipping costs.

$$D^X = \begin{Bmatrix} 0 & 1 \\ 0 & 0 \end{Bmatrix}, D^F = \begin{Bmatrix} 0 & 1 \\ 0 & 0 \end{Bmatrix}$$

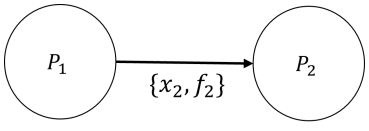


Figure 3.8: Freight Assignment and Ship Routing Problem Linkages

Figure 3.8 shows the structure of linkages between the two problems. The level of interactions between the problems is represented in the two adjacency matrices above.

The Water Tank Transport and Packing Problem [341]

This problem is a complex logistics operation involving five related problems; Facility Location, truck scheduling problem, packing problem, driver selection and vehicle routing. The problem attempts to minimise the key performance indicator of transport cost as a percentage of delivered sales value.

Problem one - Facility Location (p_1). The solution determines a set of production facilities and/or storage facilities (if the product is in stock) to make a set of products ordered by customers within a given time window. p_1 aims to minimise the total distance between a facility and delivery locations. This is subject to a set of constraints relating to delivery locations. The solution to the facility location problem constrains an instance of the truck scheduling problem and the driver selection problem.

Problem two - Truck Scheduling (p_2). This is a scheduling optimisation problem. An instance of the truck scheduling problem is constrained by the solution of the facility location problem. The solution schedules a set of trucks for delivery trips within a time window. The aim is to minimise the cost of transportation. p_2 is constrained by truck availability and capacity. The solution to the truck scheduling problem determines an instance of the driver selection problem p_4 and the cutting and packing problem p_5.

Problem three - Cutting and Packing (p_3). The instance of this problem is determined by the solution of the truck scheduling problem. The solution determines the packing and bundling arrangement of a set of water tanks on available trucks. p_3 seeks a packing arrangement that maximises the total value of tanks in each truck. p_3 is subject to a set of constraints, including; product type, truck capacity and product dimensions. The solution of the cutting and packing problem p_3 determines an instance of the vehicle routing problem p_5.

Problem four - Driver Selection (p_4). This is an assignment problem. The problem instance is determined by the solution of the truck scheduling problem and constrained by the solution of the facility location problem p_1 and the solution of the vehicle routing problem p_5. The solution determines a set of drivers for delivery trips. The aim is to minimise total transportation costs.

Problem five - Routing (p_5). p_5 is a vehicle routing problem with time windows. The problem instance is determined by the solution of the cutting and packing problem p_3. The solution to the routing problem determines the optimal routes to travel in delivering tanks to customers. The aim is to minimise the total time travelled by the scheduled trucks. p_5 is subject to a set of constraints relating to customer orders. The solution to the truck routing problem constrains an instance of the driver selection problem p_4.

Figure 3.9 presents a visual representation of the linkages in the tank delivery problem. Three adjacency matrices below describe the behavioural changes that resulted from the linked problems.

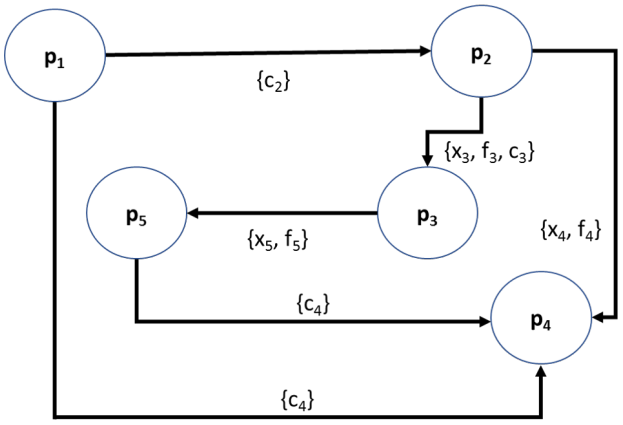


Figure 3.9: Tank Delivery Linkages

$$D^X = \begin{Bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{Bmatrix}, D^F = \begin{Bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{Bmatrix}, D^C = \begin{Bmatrix} 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{Bmatrix}$$

3.2.3 Synthetic Example From the Academic Literature

A typical example of a linked problem in the literature is the travelling thief problem (TTP) introduced by [42]. TTP is a combination of two existing problems known in the literature: the knapsack problem (KP) and the travelling salesman problem (TSP). In isolation, each problem can be represented as problem p_1 for KP defined by (x_1, f_1, c_1) and problem p_2 for TSP defined by (x_2, f_2, c_2). $x_1 \in \{0,1\}^N$, a set of binary encoding, represents the picking plan of N items at different locations, f_1 represents the objective function that maximises the total value of items picked, and c_1 represents the maximum capacity constraint of the knapsack. $x_2 = (\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_M)$ represents a permutation of M cities (i.e., a tour) and f_2 evaluates the solution set x_2 to determine the best tour (permutation) with minimum total time. Here p_2 is unconstrained and therefore, $c_2 = \emptyset$. Both problems have completely different search spaces and, therefore, can be solved differently in terms of how both problems are combined, the number of objectives used, and the design of the algorithms used [42]. The authors used an evolutionary algorithm to solve TTP in two different ways. One involves linking the two problems by combining the individual objectives to form an overall objective. The other approach involves the design of cooperation co-evolution. We present below the adjacency matrix of how each sub-problem affects each other based on the above definition of a linked problem.

$$D^X = \begin{Bmatrix} 0 & 1 \\ 0 & 0 \end{Bmatrix}, D^F = \begin{Bmatrix} 0 & 1 \\ 0 & 0 \end{Bmatrix}$$

In the above matrix, 1 indicates that solution x_1 of a given problem p_1 determines an

instance of problem p_2. In other words, a feasible instance of solution x_1 transmitted to p_2 influences the choice of solution x_2 and thereby reduces the chances of selecting an optimal solution x_2.

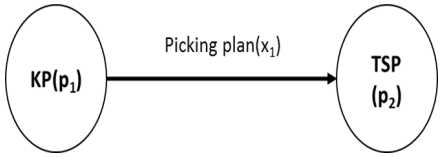


Figure 3.10: TTP Linkages

Figure 3.10 represents the structure of the problem based on two linked nodes. Here, in TTP, based on an instance of solution x_1 selected or transmitted, the attribute of x_1 (i.e., weight) increases as more items are picked on the pathway according to the picking plan in solution x_1. As more items are picked, the travel speed in p_2 decreases proportionately, thus, taking more time in p_2 to get to the next picking point (location). The impact of solution x_1 on p_2 degrades the quality of permutations of the cities x_2 (i.e., increased total time of tour).

3.3 Algorithmic Approaches and Linked Optimisation Problem Techniques

This section provides reasoning regarding algorithmic approaches implemented in solving linked problems in a supply chain, using the examples provided in Section 3.2. This enables us to provide a general formulation of algorithmic approaches that incorporates the level of interaction among the supply chain decision variables.

3.3.1 Existing Objective Function Formulation for Linked Problem

Our motivation here is to provide precisely generic mathematical expressions for objective function formulation in existing linked optimisation problem papers. Therefore, we intend to give the following definition to describe the peculiarity of linked problems. Our observations from the examples presented in Section 3.2 will help us to ascertain these peculiarities and the appropriate formulation of the functions in terms of D.

We describe the objective function as follows;

$$\begin{aligned} &min/max\big(f^1(x_*^1),\cdots,f^n(x_*^n)\big)\\ &s.t. \quad x_*^i \in x^i \quad and \quad i=1,\cdots,n \end{aligned}$$

where x_*^i is a feasible solution to problem p_i, x^i is the i-th set of solutions, and f^i is the i-th objective function in the objective space.

Derivation of Objective Function for Linked Optimisation Problem

The objective function for a linked problem is formulated in different ways. The formulation can incorporate two or more objective values corresponding to the sub-problems. This depends on the number of problems in the linked problem network. Specifically, the formulation may contain functions with different measurement units and are often computed as a single-valued function. The single-valued function can be expressed in different forms, such as cost value, profit value, production quantity or processing time. To the best of our knowledge, the objective function for a linked problem can be formulated as an aggregation of n objective functions or expressed as an exclusive objective function.

- Aggregation. A linked problem can be formulated as an aggregation of objective functions by summing the values of the objective function f^i corresponding to each problem p_i into an aggregated value. Aggregation is quite similar to multi-objective optimisation. Multiple objectives are combined into a single objective scalar function, often known as weighted-sum or scalarisation [59]. This formulation is possible if all the corresponding functions have similar measurements. However, in most cases, these values have different measurement representations. Thus, new parameters (weighting coefficients) are introduced to convert the individual objective values to a consistent measurement unit that allows aggregation. These weighting coefficients may not be directly associated with the corresponding objective functions [59]. In papers [73] [228] [242] [341], the n objective functions corresponding to individual problems are aggregated to form a single holistic value. Here, we express the aggregated functions as follows;

$$min \sum_{i=1}^{n} f^i(x_*^i).\alpha_i$$

  where;

  - $f^i(x_*^i)$ is the objective value of the optimal solution of an i-th problem.
  - n is the number of problems in P.
  - α_i is a constant value (weighting coefficient) associated with i-th problem p_i that converts the objective value of p_i to a single measurement unit equivalent to the objective values of other problems in the linked structure.

- Exclusive. Here, a single function is picked as a representative function. This formulation is proposed by [70] for multi-objective optimisation and is known as ε-constraints. Here, a decision maker chooses one objective out of n to be optimised, and the remaining objectives are constrained to given target values. The choice of objective function has no bearing on whether the problem is a parent or a child, while

other functions are used as constraint functions. See papers [356] [375] [240] for more details.

$$
\begin{aligned}
& min/max \quad f^i(x_*^i) \\
& x_*^i \in X \quad and \quad i = 1, \cdots, n \\
& s.t. \begin{cases} \{f^1(x_*^1), \cdots, f^n(x_*^n)\} \setminus f^i(x_*^i) \\ s.t. \quad x_*^i \in X \quad and \quad i = 1, \cdots, n \end{cases}
\end{aligned}
$$

A special exclusive function case is seen in the bi-level optimisation problem (BOP). In BOP, the objective function induces a hierarchy between problems p_1 and p_2.

- A leader p_1, which selects a feasible solution x_*^1 and tries to optimise f^1.
- A follower p_2, which searches a feasible solution x_*^2 to optimise f^2.

See more details about BOP in paper [232].

3.3.2 Algorithmic Approach for Linked Optimisation Problem

It is essential to provide a general overview to address a linked problem. Several algorithms have been designed in the literature to tackle linked optimisation problems. We will consider some of the approaches used in the examples provided in section 3.2.

We identified three major approaches to solving linked optimisation problem P; sequential, concatenation and combined approaches. Let S represent a sequential approach, T represent a concatenation approach, and Z represent a combined approach.

Sequential Approach (S)

The sequential algorithmic approach solves a linked problem P in sequence and improves the solutions of each problem p_i. This approach is commonly known for solving linked problems in a hierarchical structure usually between two problems (decision makers), i.e, p_1 and p_2 [232] [182]. An example of such a linked problem is seen in a bi-level optimisation problem. See [346]. We express a sequential approach as $S\big(A_1, \cdots, A_n | p_1, \cdots, p_n\big)$, where $\{A_1, \cdots, A_n\}$ represents a set of n algorithms used in S and $\{p_1, \cdots, p_n\}$ represents a set of n problems in P. Each algorithm A_i represents the i-th algorithm that runs on the i-th problem in P. See detailed definition of P in section 3.2.1. Algorithm 3 shows a sequential approach for solving a linked problem with given $n \geq 2$. A_i solves problem p_i based on solution x_*^{i-1} and obtains solution x_*^i. Then, feeds its solution x_*^i to the next algorithm A_{i+1} to solve problem p_{i+1}. For more details, see [73] [182] [228] [240] [356].

Algorithm 3: Sequential Algorithmic Approach S

initialisation ;
new candidate solution $(x_*^1) \leftarrow \mathrm{A}_1|p_1$;
for *each i from 2 to n* **do**
  new candidate solution $(x_*^i) \leftarrow \mathrm{A}_i|(p_i, x_*^{i-1})$;
end
Result: return overall best solution $(x_*^1, \cdots, x_*^n)$

Where $2 \leq i \leq n$.

A sequential approach can be adopted iteratively in a cyclic linked structure. Each problem is solved in sequence, and the solution x_*^i for each problem p_i is used to improve the next solution iteratively until a stopping criterion is met. Algorithm 4 shows an iterative sequential algorithmic approach for a linked problem with $n \geq 2$. The machine scheduling and vehicle routing problem are typical examples that adopted the iterative sequential algorithmic method.

Algorithm 4: Iterative Sequential Algorithmic Approach S_I

initialisation ;
new candidate solution $(x_*^1) \leftarrow \mathrm{A}_1|p_1$;
while *Stopping criterion not met* **do**
  for *each i from 2 to n* **do**
    new candidate solution $(x_*^i) \leftarrow \mathrm{A}_i|(p_i, x_*^{i-1})$;
  end
  evaluate $(x_*^1, \cdots, x_*^n)$;
  keep best overall solution;
  new candidate solution $(x_*^1) \leftarrow \mathrm{A}_1|(p_1, x_*^n)$;
end
Result: return overall best solution $(x_*^1, \cdots, x_*^n)$

Concatenation

The concatenation algorithmic approach solves linked problems as a whole. We express a concatenation approach as $T\big(A|(p_1, \cdots, p_n)\big)$, where A denotes an algorithm and $\{p_1, \cdots, p_n\}$ is a set of problems associated with P. This approach constructs a solution mix of problems $\{p_1, \cdots, p_n\}$ as a single solution representation. The approach uses specially designed operators to enhance the algorithm performance further to explore complex search spaces relating to the linked problem [250]. Algorithm 5 shows the steps used by the concatenation approach for a linked problem with $n \geq 2$. Algorithm A generates a mix of candidate solutions to problems $p_1, \cdots, p_n$ and evaluates the mix of solutions. The algorithm uses special operators to explore the search space to improve the solution mix until a stopping criterion

is met and then returns the best overall solution mix $(x_*^1, \cdots, x_*^n)$. See [242] [356] [375] for more details of the application of concatenation approach.

Algorithm 5: Concatenation T

```
initialisation ;
initial solutions(x_*^1, ..., x_*^n) ← A|(p_1, ..., p_n);
while stopping criterion not met do
    evaluate (x_*^1, ..., x_*^n);
    keep best overall solution;
    generate new (x_*^1, ..., x_*^n) with A operators;
end
Result: return best overall solution mix
```

The concatenation approach was adopted in cooperative coevolutionary approach. This approach runs the algorithms corresponding to each problem in parallel and allows cooperation between the problems while evaluating individual solutions [182]. Solutions from one problem are evaluated based on their performance when combined with a representative solution from another problem and vice versa [250]. See an example of job-shop scheduling and vehicle routing problem in [182].

Algorithm 6 shows the pseudo-code of cooperative coevolutionary approach for solving a linked problem with $n \geq 2$. The approach combines n corresponding EAs to generate n sub-populations $pop^1, \cdots, pop^n$, one for each problem $p_1, \cdots, p_n$ and tries to improve them independently while periodically exchanging information to keep an overall view of the linked problem [232]. The different sub-populations $pop^1, \cdots, pop^n$ are used to build complete solutions and are evaluated.

Combined

The combined approach uses iterative and concatenation algorithmic processes to solve a more complex linked optimisation problem. The approach can use a mix of the two approaches (i.e., Iterative I and Concatenation T) to solve n complex interdependent problems, usually with $n > 2$. We describe combined approach as $Z(I, T)$.

In Table 3.1, we provide a classification of algorithmic approaches adopted for solving linked optimisation problems. The information in the table uses the examples we presented in section 3.2.

3.4 Holistic Optimisation of Linked Optimisation Problems

The section will describe the importance of holistic optimisation to a linked problem and the need to develop tools to solve them.

Algorithm 6: Cooperative Coevolutionary T_{cc}

initialisation ;
for *each i from 1 to n* **do**
  initial population $(pop^i) \leftarrow A_i|p_i$;
end
while *Stopping criterion not met* **do**
  evaluate the fitness of each solution from $pop^1, \cdots, pop^n$;
  select m size of $pop^1, \cdots, pop^2$;
  for *each j from 0 to m* **do**
    evaluate $pop^1[j], \cdots, pop^n[j]$;
    keep overall best solution $x_*^1, \cdots, x_*^n$;
  end
  for *each i from 1 to n* **do**
    generate new population $(pop^i) \leftarrow A_i|p_i$;
  end
end
Result: return overall best solution $x_*^1, \cdots, x_*^n$

Table 3.1: Classification of Algorithmic Approaches for Linked Optimisation Problems

Linked Problems	Algorithmic Approaches	D^X	D^F	D^C
Combining Lot Sizing & Vehicle Routing Problem [228]	Sequential	✓	✓	
Integrated Machine Scheduling & Vehicle Routing [356]	Sequential & Concatenation			✓
Workforce Scheduling & Routing Problems [375]	Concatenation			✓
Crane Handling Scheduling & Truck Routing Problem [73]	Iterative Sequential	✓	✓	
Order Batching, Sequencing & Routing Problem [75]	Combined	✓	✓	
Solving Location, Allocation, & Capacity Planning [240]	Sequential	✓	✓	✓
Freight Assignment & Routing Problem [242]	Concatenation			✓
The Water Tank Transport & Packing Problem [341]	Combined	✓	✓	✓

3.4.1 Holistic Optimisation

In any linked optimisation problem network, the focus of each network node is the optimisation of a specific component problem. Parts of the network may collaborate to co-optimise, and various techniques exist to study that: multi-objective and bi-level optimisation. However, the network considered as a system has effects on its surrounding environment, which are not captured in the component problem definitions and may have no relation to the component optimisation goals. Examples: carbon emissions of an industrial supply chain, e.g. offshore platform supply; health effects on consumers of optimised food manufacturing; group company corporate objectives not represented as subsidiary company objectives.

We, therefore, introduce the concept of holistic goals for the linked problem that are distinct from the individual component goal functions. With a given vector of solutions $x = (x_*^1, ..., x_*^n)$, these are represented as a holistic optimisation goal $H(x_*^1, ..., x_*^n)$ where the x_*^i are the solutions chosen for the component problems. H is a function of the vector of solutions x. The aggregations or joint solutions in Sections 3.3 are derived from the objective

functions of the individual problems. In general, H may be a multi-valued vector of objectives, but the main point is that it is independent of the decision-making processes in the system. Holistic optimisation is, therefore, defined as an external optimisation by regulators external to a linked optimisation problem. Defining a holistic optimisation goal is essential due to how the world operates, and many environmental problems we have are due to our failure to account for these regulators within our internal systems.

External Regulators

Many industries lack environmental sustainability practice in their supply chains [342]. Consequently, business activities have continued to widely threaten the environment through the disposal of pollutants such as toxic materials, carbon, and sulphur emissions [78]. Hence, this has compelled government and non-government organisations to enhance environmental sustainability through regulations and legislation [342] as minimising such environmental damage is critical to today's supply chains [110].

These regulations are external systems that might influence and regulate the individual objectives of internal systems. The regulations are levers available to the external regulator. Therefore, we postulate an external regulator aiming to optimise H. The solution vector $x = (x^1_*, ..., x^n_*)$ chosen via the linked optimisation is determined by the sets (x^i, f^i, c^i) and the matrices showing the interactions between them. We assume the regulator can affect X, f, C using some set L of control levers. The holistic optimisation problem is for the regulator to select control levers that optimise H. Imposing these levers will provide a way to understand the knock-on effects in complex systems. Game theory might be a way to think about how the control levers affect the decision-making by changing the game.

Game theory

Game theory studies strategic interactions and decision-making between agents based on a mathematical framework [50]. The concept of game theory can be used to explore how the decision-making process can be understood externally in terms of the behaviour of an internal system in different situations. The idea of a game theory in holistic optimisation is to think of how changing the rules of a multi-player game affects the payoff of the entire system (game). Here, the system payoff is represented by the payoff of the individual player of the system (game).

Game theory allows multi-players to make sequential decisions from a set of rules. These rules are decisions made by regulators that trigger appropriate levels of reactions from the system. From the regulator's viewpoint, we want to investigate how the individual players react to the range of strategies available to the opponents. Here, a strategy is a list of moves that maximises a payoff, given the other player's moves.

In the system, we then apply an optimisation algorithm to determine the players' strategic choices given what the regulator does. We can then model the optimisation system using

appropriate algorithms to get an outcome based on the given strategic choices and the regulator's rules/decisions. These algorithms will serve as a surrogate for strategic choices of the individual players. By running the algorithms, we get a set of solutions (choices) and apply a holistic optimisation goal H to the set of solutions. The process will involve a simulation of strategic choices over a given set of decisions made by the regulators to investigate what choices and regulator's rules can obtain equilibrium points.

3.4.2 Motivation for Conceptual Framework

Interestingly, linked problem patterns are recognised in the literature. This chapter has extracted the concept of a linked problem from the literature and developed a formalism to describe it. However, its conceptual framework has not been developed. In light of future directions, it is imperative that we develop a framework to systematically study linked problems as a formal topic in optimisation problems. The need for a formal framework is to seek holistic treatment for obtaining an optimal solution, such that the underlying dependencies among the sub-problems are considered.

It is essential to think about a linked problem in terms of a holistic goal rather than individually optimising each sub-problem. This will allow the individual sub-problems to collaborate or compete in a certain way. We, therefore, expect that the framework should provide a way to solve a linked problem in terms of holistic goal H. There should be an intellectual method of analysing external features underlying the holistic optimisation and defining the external control in terms of the internal objectives. We need people to start looking at a holistic linked problem and developing tools to solve them.

To the best of our knowledge, no complex computational analysis has provided insights into the interactions between the different sub-problems [41]. Therefore, a formal, conceptual framework is required. In addition, the theoretical investigation of computational methods for linked problems poses a new challenge, requiring an appropriate framework.

3.5 Key Research Directions

This section provides the key research directions of holistic linked optimisation. Developing a framework that seeks to solve linked optimisation problems in terms of holistic goals must answer key research questions. These research questions are important in delivering and guiding our understanding of the extent of the underlying holistic linked optimisation problem.

- How can we use the framework to state what the holistic linked optimisation problem is? It is important to note that a clear understanding of a holistic linked problem requires a succinct framework. This will create opportunities for exploring problems from different perspectives as a whole.

- How would the framework be used to solve the holistic goal H? Developing an appropriate holistic framework will provide the research community with the need to look at linked optimisation problems in terms of holistic measures rather than individual goals.
- How should regulators use the framework to think about H? The study of external measures on holistic goals can provide opportunities for determining the extent to which complex systems are controlled by external influence. In the context of an appropriate holistic framework, accounting for the imposing external levers, such as environmental sustainability, would likely raise some questions on what treatments can be applied or determined.
- What work/techniques already exist? Adopting a framework for holistic measures can be geared toward identifying the suitability of existing techniques that can be used for holistic treatments and how they can be managed consistently across different levels of complexity.
- What are the gaps? A study into a holistic framework of a linked optimisation problem is an opportunity to identify potential areas of research that have not been considered. The framework should allow people to start thinking of developing tools/solutions not already available for holistic treatment of a linked optimisation problem.
- How can the framework be used for modelling complex environmental and socio-linked problems? This question seeks to identify what and how external features can be used as part of internal goals to define holistic treatment for the linked optimisation problem.

This suggests areas for further study.

- Holistic decision support approaches that simulate the optimisation network from observed data and seek control level settings that optimise H. Holistic optimisation is our main goal for further research work.
- Game-theoretic or other models of the linked optimisation network that allows simulation of the network response to control levers. Game theory seems to be a promising technology to achieve our goal.
- Benchmark problems capturing problem networks seen in literature and real-world problems. This is an important component in creating a platform for investigation.

3.6 Summary

This chapter's contribution addresses how to develop a formalism for linked optimisation problems. It addresses the research question (RQ1) and relates to objective (O1). This chap-

ter provides a rationale for identifying, defining and improving the understanding of linked optimisation problems. The chapter extracted the concept of a linked problem from the literature to develop a formal definition to describe it. It was observed that the study of linked systems requires an appropriate framework that identifies linkages in the data and the processes that cut across the linked problem components. This then introduced the concept of holistic optimisation goals, which are distinct from the individual component goal function as an external optimisation by regulators external to the linked problem. The introduction of the external features raises several research questions, which will need to addressed as suggested for further research directions. Furthermore, this chapter suggests a need for a systematic analysis of a linked problem system that provides an understanding of the interactions among the individual problems. The next chapter addresses the linkages of data as a prerequisite to understanding linked optimisation problems.

4

Supply Chain Data Linkages

The integration of several optimisation problems is largely influence by the data which are propagated from the decisions made from the individual components of the linked system. This chapter solely addresses how linkages of linked systems are identified from a functional perspective. The study of data linkages in this chapter is to serve as a prerequisite for identifying linkages in the components of a linked system.

4.1 Introduction

Supply chain data are one of the most critical assets in the economy of the 21st century. The entire supply chain industry rely on and are centred around exploiting large data sets, as many modern supply chain processes generate millions or even billions of data records daily stored in databases. Understanding the relationship between data and gaining insight from data is central to their commercial success.

A user such as a business analyst may gain access to an existing supply chain database. However, expertise about how data is structured and how data tables relate to each other may not be provided, and little or no documentation exists. It could be that the technical and domain experts have moved on or left the business altogether, or many different groups have contributed to the database over time without a single authority fully understanding the overall information about the supply chain processes. This is a significant roadblock to exploiting this data. This challenge has mainly been addressed through highly time-intensive human analysis and exploration by domain experts [100, 109, 198, 319]. However, such an approach is limited by time, cost and amount of information to explore. Furthermore, the approach is likely to be error-prone [26, 111, 161]. This suggests a need for an automated mechanism to speed up the supply chain data discovery process.

In this chapter, we investigate several relationship discovery algorithms that infer links between columns of tables and propose a framework that combines them into an overall framework. To the best of our knowledge, several approaches have been proposed to determine semantic relationships between database schemas. In addition, several variations have been reviewed, each with its strengths and weaknesses. These variations can be found

in [7]. However, little research has been seen exploring various ensemble strategies for combining several relationship discovery algorithms. One of these strategies was seen in [255], which is in the space of schema matching. The strategy focuses on the manipulation of database schema elements for mapping [306].

Our motivation for combining several algorithms is to reduce the generalisation error of the prediction produced by the individual algorithms [219]. Individual algorithms are diverse and independent, so the predictions made by a single algorithm may lead to imperfect discovery compared to a framework that combines several approaches [306].

Our proposed approach emphasises recall, and this is based on the premise that our methods discover different relationship types; primary/foreign key (explicit) and semantically equivalent (implicit) relationships. We only rely on the explicitly defined primary key/foreign key relationship as our gold standard. Thus, false positives (which are more likely to be semantically equivalent relationships) could be discovered due to the impact of the specified gold standard. This chapter makes the following contributions;

- We investigate the problem of automatically discovering primary and foreign keys and semantically equivalent (implicit) relationships by ensemble methods.
- We use hierarchical clustering method as an ensemble framework to combine the prediction of individual discovery algorithms to provide a comprehensive matching outcome.

The rest of this chapter is structured as follows. Section 4.2 briefly explores some related work in relationship discovery. Section 4.3 defines the problem and describes the individual algorithms and their ensemble strategies. The experimental evaluations are provided in Section 4.4. Finally, a summary is given in Section 4.5.

4.2 Related Work

Several approaches have been proposed in the literature using different categories of data. For instance, Jiang and Naumann [198] proposed a holistic discovery of both primary key and foreign key (HoPF) as a subset of sets of unique column combinations and inclusion dependencies based on score function and several pruning rules. Rostin et al. [319] proposed ten feature-based approaches to automatically detect foreign keys using a machine learning model. In [109], K-Means clustering was used to solve a multi-schema matching problem. They used a well-known term frequency-inverse document frequency (TFIDF) weighting to convert attributes to points in a vector space model and used cosine measure as a distance metric between attributes. Mehdi et al. [263] proposed a content-based matching approach to determine the relationship between attributes which rely on the combined strength of Google as a web semantic and regular expression as pattern recognition. [397] proposed an

unsupervised solution that clusters a set of columns to identify attribute relationships based on similar value characteristics using Earth Mover's Distance (EMD) as distance measures.

4.3 Ensemble-Based Discovery

4.3.1 Problem Definition

For a given database of n tables, $T = \{t_1, t_2, \cdots, t_n\}$, let $C = \{c_1, c_2, \cdots, c_\theta\}$ be the set of all columns of tables T where θ is the number of columns in the database. We define $t_i(c_i)$ as a table with an associated column where c_i is an i-th column of table t_i.
Let $\Delta = \{(c_i, c_j) : \exists \quad t_i(c_i) = t_i(c_j), (c_i, c_j) \in C \times C\}$ be a set of column pairs (c_i, c_j) of the same table. We define $g_k = (C_k, E_k)$ as a graph of inferred relationships between set of columns (nodes) C_k and $E_k \subseteq C_k \times C_k \subset (C \times C) \setminus \Delta$ as the set of edges of g_k. Columns c_i and c_j are nodes in C_k, and each pair of columns (c_i, c_j) represents an edge in E_k, such that $(c_i, c_j) \in C_k \times C_k$. Let $f_k : C \times C \rightarrow g_k$ be a given discovery algorithm that produces graph g_k.

Our task is to determine the relationships between database tables which forms a graph G. The relationships include primary/foreign key and semantic relationships, which are determined by different discovery techniques to produce graphs. The graphs are combined, with appropriate ensemble methods, to produce a global graph. The discovery techniques exploit metadata/schema information and column values in the relational database model.

1. Input Parameters
    a) C - A set of all columns of the tables in T in the database DB.
    b) f_k – A suitable method for discovering table relationships.
2. Output Parameters
    a) $g_k = (C_k, E_k)$ - A graph containing a set of column pairs (c_i, c_j) in C_k where $C_k \in C$.

4.3.2 Relationship Discovery Algorithms

This section will introduce a number of related relationship discovery algorithms (i.e, pseudo-primary key discovery, name similarity, usage-based approach, cosine similarity, semantic similarity in a taxonomy, soundex similarity, value range similarity, and content-based similarity) which provide different alternative strategies for automatically identifying relationships between different database tables using different categories of database information.

Pseudo-Primary Key Discovery (Pri)

Pri is important in an application area, where no explicit definition of primary and foreign key constraints is available [319]. Existing work in this area can be found in [198, 289, 319,

396]. We denote A as the subset of C, $A \subseteq C$, which contains all columns with explicitly defined primary key columns in a database. Let B be defined as the subset of $C \setminus A$, $B \subseteq C \setminus A$, which are columns qualified as potential primary key candidates. Sets A and B do not share any columns. Let X be the union of A and B: $X = A \cup B$. We then calculate a graph g_k in which the nodes are columns from X plus their associated foreign key columns. Two column nodes are linked in the graph if they are in a primary/foreign key candidate relationship. We use the following four tests to infer B.

- Alphanumeric Datatypes Test: Columns with alphanumeric datatypes.
- Nullability Test: Non-null columns.
- Uniqueness Test: Columns with unique values.
- Word Character Test: Columns with letter, digit, or underscore character.

We distinguish two cases in primary key/foreign key column pairs:

- Either a column is explicitly marked as a foreign key in the database itself,
- Or we need to establish that the second (foreign key) column only contains values that appear in the first (primary key candidate) column.

In Equation 4.1, $values(c_i)$ denotes values in column c_i. $w(c_i, c_j)$ returns 1 if two columns c_i and c_j are in a (potential) primary / foreign key relationship, and 0 otherwise:

$$w(c_i, c_j) = \begin{cases} 1, & \text{if } c_i \in A \text{ and } c_j \text{ is foreign key for } c_i \text{ and } t_i(c_i) \neq t_i(c_j) \\ 1, & \text{if } c_i \in B \text{ and } values(c_j) \subset values(c_i) \text{ and } t_i(c_i) \neq t_i(c_j) \\ 0, & \text{otherwise} \end{cases} \tag{4.1}$$

Name Similarity (NSim)

Nsim is used to determine the linkages between tables by identifying the similarity between column names associated with each table. Several names used in identifying tables and columns are usually designated based on the nature of the business activities. Thus, column names may have inconsistent designations across tables. For instance, a column name "Customer Name", might be represented either as "CustName", "CustomerN" or "CstName". We used Jaro-Winkler ($JWinkler(c_i, c_j)$) to discover the similarity between two column names (c_i and c_j) because it is a well-known algorithm used as far back as the 80s. This is currently used in name similarity matching like entity matching [372]. See [193] and [374] for detailed mathematical definitions. We used java-string-similarity [1] library for our implementation. In Equation 4.2, we define the $Score(c_i, c_j)$ function for all threshold

[1] https://github.com/tdebatty/java-string-similarity

dependent algorithms. The $Score(c_i, c_j)$ function returns 1 if a given $Metric$ function, like $JWinkler(c_i, c_j)$, produces a value greater than or equal to a given $Threshold$ and if c_i and c_j are not from the same table t_i. $Score(c_i, c_j)$ returns 0 otherwise. In the NSim algorithm, we implement $JWinkler(c_i, c_j)$ as the $Metric$ function. The value of $JWinkler(c_i, c_j)$ is a real number between the range of 0 and 1. If this value is greater than or equal to the $Threshold$, 1 is assigned to $Score(c_i, c_j)$. This allows us to add the two columns as nodes to graph g_k and connect them in the graph.

$$Score(c_i, c_j) = \begin{cases} 1, & \text{if } Metric \geq Threshold \text{ and } t_i(c_i) \neq t_i(c_j) \\ 0, & \text{otherwise} \end{cases} \tag{4.2}$$

Usage-Based Approach (Usage)

Usage uses a set of existing database scripts to infer relationships between tables. Scripts may include existing database logic such as procedures, functions, views or user queries. From these scripts, we extract all pairs of columns that co-occur in linking tables together. This approach was first introduced in [122]. The usage-based approach is suitable in special cases where column names are opaque, or there is no sufficient schema and data instance information. However, it is often difficult to obtain suitable usage data [305]. We used General SQL Parser (GSP) library [2] to implement this approach. Let $S = \{s_1, \cdots, s_q\}$ be the set of existing scripts for a database. s_i denotes a single script and references a set of tables T_{s_i} in its logic. We define $T_{s_i} = \{t_{s_i 1}, \cdots, t_{s_i \iota}\}$, where ι is the number of tables in T_{s_i}. If script s_i contains a link statement, e.g. a join statement, between tables $t_{s_i x}$ and $t_{s_i y}$, and more specifically links the referenced columns in $t_{s_i x}$ and $t_{s_i y}$ respectively, we then, infer a link between those two columns and add the two columns as nodes to graph g_k.

Cosine Similarity Approach (Cosine)

Cosine uses vector representation to measure the cosine angle between two vectors. Cosine was used in [109] as a distance metric measure for clustering attributes. We adopt cosine similarity to represent each attribute/column as a vector using Term Frequency Inverse Document Frequency (TFIDF) weighting computation. TFIDF is a term weighting scheme for cosine computation. TFIDF is a product of a term frequency (TF) weight factor and an inverse document frequency (IDF) weight factor. We define the cosine similarity metric between a pair of columns as $CoSim(c_i, c_j)$. See detailed computation of cosine similarity $CoSim(c_i, c_j)$ in [328]. The cosine similarity value $CoSim(c_i, c_j)$ is a real number between 0 and 1 and it represents the $Metric$ function defined in equation 4.2. If the value is greater than or equal to the $Threshold$ in equation 4.2, we then assign 1 to $Score(c_i, c_j)$ or 0 otherwise. A $Score(c_i, c_j)$ of 1 will add the two columns as nodes to graph g_k and connect them in the graph.

[2] http:/dpriver/www..com/

Semantic Similarity in a Taxonomy (Sem)

Sem exploits additional external information to measure the similarity between a pair of words or concepts. The key resource used is a knowledge-based database, such as a business-specific ontology or a general-purpose database like WordNet [270], which encodes relations between concepts. For example, when column headers are described slightly differently, e.g., "AUTOMOBILE_NO" can conceptually mean the same as "VEHICLE_ID". We used a knowledge-based function in [269] to measure the similarity between a pair of columns. We define the knowledge metric as $Sem(c_i, c_j)$, which computes the average similarity score by combining resultant similarity scores of substrings of c_i and c_j. We define v_{ik} as the $k-th$ substring / term associated with the name for column c_i. The $SemSim(v_{ik}, v_{jk})$ metric in equation 4.3 is used in the $Sem(c_i, c_j)$ metric computation (see [269]) which returns a similarity score between a pair of terms v_{ik} and v_{jk} associated with the names of columns c_i and c_j respectively. A stopword (i.e., most common word in a language) term returns a score of 0. If both terms are not in the knowledge networks, name similarity $JWinkler(v_{ik}, v_{jk})$ is used. $JWinkler(v_{ik}, v_{jk})$ is also used for terms of adjectives or adverbs in the knowledge network. Lastly, if the pair of terms are both verbs or nouns in the knowledge networks, we then compute $sim_{Lin}(v_{ik}, v_{jk})$, otherwise score returns 0.

$$SemSim(v_{ik}, v_{jk}) = \begin{cases} 0, & \text{if } v_{ik} \text{ or } v_{jk} = stopword \\ JWinkler(v_{ik}, v_{jk}), & \text{if } v_{ik} \text{ or } v_{jk} \notin ontologies \\ sim_{Lin}(v_{ik}, v_{jk}), & \text{if } v_{ik} \text{ and } v_{jk} \in ontologies(noun) \\ sim_{Lin}(v_{ik}, v_{jk}), & \text{if } v_{ik} \text{ and } v_{jk} \in ontologies(verb) \\ JWinkler(v_{ik}, v_{jk}), & \text{if } v_{ik} \text{ or } v_{jk} \in ontologies(adv) \\ JWinkler(v_{ik}, v_{jk}), & \text{if } v_{ik} \text{ or } v_{jk} \in ontologies(adj) \\ 0, & \text{otherwise} \end{cases} \tag{4.3}$$

We implemented $sim_{Lin}(v_{ik}, v_{jk})$ using Semantic Measures library [3]. See computation in [241]. It takes two concepts and returns their semantic relatedness value. Let $Sem(c_i, c_j)$ represents the $Metric$ function in equation 4.2. The $Score(c_i, c_j)$ function defined in equation 4.2 is assigned 1 if the $Sem(c_i, c_j)$ is greater than or equal to the $Threshold$ and if the column pair are not from the same table.

Soundex Similarity (Soundex)

It is a phonetic algorithm that indexes a string by sound in English. It simply evaluates the letters of a string and assigns a numeric value. Soundex is used to identify the relationship between two tables based on the phonetic similarity between their column names. See computation in [299]. We implement Soundex using Apache Commons library [4] in Java.

[3] https://www.semantic-measures-library.org/sml/index.php?q=downloadssml

[4] https://commons.apache.org/proper/commons-codec/download$_codec.cgi$

We denote the phonetic similarity as $Sdex(c_i, c_j)$. The value of $Sdex(c_i, c_j)$ is between 0 and 4. A $Sdex(c_i, c_j)$ value of 4 means that a pair of column names sound strongly similar, and 0 means otherwise. $Sdex(c_i, c_j)$ computes the $Metric$ value in equation (2) to assign $Score(c_i, c_j)$ a score 0 or 1.

Value Ranges Similarity (Val)

Val uses the minimum and maximum values of column pairs to determine whether they are linked. Val works with numeric, strings or date datatypes. Two columns of the same datatype are similar if they have a similar value range pattern. We denote a_i as a pair of minimum and maximum values $\langle min(c_i), max(c_i)\rangle$ for column c_i. Columns c_i and c_j are logically equivalent ($a_i \equiv a_j$), if a_i is similar to a_j or vice versa. The check $range(c_i, c_j)$ in equation 4.4 returns 1 for similar value ranges between two columns c_i and c_j or 0 otherwise. (c_i, c_j) is added to g_k if $range(c_i, c_j)$ is 1.

$$range(c_i, c_j) = \begin{cases} 0, & \text{if } datatype(c_i) \neq datatype(c_j) \text{ or } t_i(c_i) = t_i(c_j) \\ 1, & \text{if } a_i \equiv a_j \text{ and } t_i(c_i) \neq t_i(c_j) \\ 0, & \text{otherwise} \end{cases} \tag{4.4}$$

Content-Based Similarity (Col)

Col exploits and compares data instances to determine the relationship between columns pair. $content(c_i, c_j)$ returns 1 if the set of value samples in column c_j is a subset of unique values of column c_i, and 0 otherwise. (c_i, c_j) is added to graph g_k if $content(c_i, c_j)$ is 1.

$$content(c_i, c_j) = \begin{cases} 0, & \text{if } datatype(c_i) \neq datatype(c_j) \text{ or } t_i(c_i) = t_i(c_j) \\ 1, & \text{if } samplevalues(c_j) \subset values(c_i) \text{ and } t_i(c_i) \neq t_i(c_j) \\ 0, & \text{otherwise} \end{cases} \tag{4.5}$$

4.3.3 Ensemble Strategies

We used a voting scheme and hierarchical clustering to find the best combination of graphs generated by the discovery algorithms.

Voting Scheme

The voting scheme checks if the individual graphs share common edges. It uses a weighting measure to determine the proportion of graphs that contain a pair of columns (c_i, c_j). Given, g_k and $P_{weighting}$, we can generate a global graph G. We defined w_k in equation 4.6 as a score that indicates whether a pair of columns (c_i, c_j) exists in graph g_k. w_k returns 1 if a pair of columns (c_i, c_j) is an element of $E_k \in g_k$ and 0 otherwise. We compute the weighted

value of $p_{(c_i,c_j)}$ in equation 4.7 for each pair of columns (c_i, c_j) as the sum of scores of w_k divided by the number of graphs m. We then generate a global graph G by adding a pair of columns (c_i, c_j) to graph G where the obtained weighting value of $p_{(c_i,c_j)}$ is equal or greater than a given $P_{weighting}$.

$$w_k = \begin{cases} 1, & \text{if } (c_i, c_j) \in E_k \\ 0, & Otherwise \end{cases} \tag{4.6}$$

$$p_{(c_i,c_j)} = \frac{\sum_{k=1}^{m} w_k}{m} \tag{4.7}$$

Hierarchical Clustering

We used the clustering approach (hierarchical clustering) proposed in [71] to group variables strongly related to each other into homogeneous clusters. Each variable represents a graph g_k. We used hierarchical clustering proposed in [71] to group the variables (graphs) into clusters based on how they are strongly linked. See [71] for a detailed formulation of the hierarchical clustering method proposed in the study.

The rationale for this strategy is that members in each cluster contain similar prediction patterns. We can, therefore, select a member of each cluster and combine it with a selected member of another cluster to exploit diversity and reduce error in prediction.

We represent each graph g_k as a categorical variable Φ_k and we defined $\{\Phi_1, \cdots, \Phi_m\}$ as a set of Φ categorical variables where $\Phi_k \in \Phi$ and $k = 1, \cdots, m$. m is denoted as the total number of variables (number of graphs). Then, let x be a set of all pairs of columns in $(C \times C) \backslash \Delta$, such that $E_k \subset x$. Φ_k has the same dimension (number of column pairs) as x, and for each variable Φ_k, contains binary strings of 0 and 1. String 1 indicates that $(c_i, c_j) \in E_k$ and 0 otherwise.

Let $\mathscr{P} = (\mathscr{P}_1, \ldots, \mathscr{P}_q)$ be a partition into q clusters of Φ variables. q denotes the total number of clusters, and $\mathscr{P}_l$ is the $l-th$ cluster of $\mathscr{P}$.

We generate $\{G_1, \ldots, G_\alpha\}$ as a set of graphs $\mathfrak{G}$ where G_i is the ith graph in $\mathfrak{G}$. We expect to obtain at least a graph from $\mathfrak{G}$ graphs which give a strong and improved relationship prediction between column pairs. We denote α as the total number of graphs (i.e., the number of possible combinations of variables from each cluster). This is expressed in the equation below;

$$\alpha = \prod_{l=1}^{q} |\mathscr{P}_l|$$

$|\mathscr{P}_l|$ denotes the number of Φ variables in cluster $\mathscr{P}_l$. Let Φ_{jl} be a variable in cluster $\mathscr{P}_l$, so that each graph $G_i \in \mathfrak{G}$ is produced by combining a set of q variables selected from each cluster using intersection operation. This is expressed below as follows;

$$G_i = \bigcap_{l=1}^{q} \Phi_{jl}$$

4.4 Experimental Evaluation

4.4.1 Dataset description

The two datasets (TPCH[5] and AdvWork[6]) used for this chapter are supply chain synthetic datasets which have been used in the literature. For ease of comparison, the TPCH used the same parameter setting used in [198]. We stored the individual datasets in an Oracle database. The characteristics of the two datasets are given in Table 4.1. Both synthetic datasets contain database views and procedures we used as existing database queries for the usage-based approach. The state-of-the-art algorithms selected for comparison in this chapter have been tested on these datasets.

Table 4.1: Data Characteristics

Data	No of Tables	No of Columns	AvgNo of Columns per Table	MaxNo of Columns per Table	Total Rows	No of Queries	Primary keys	Foreign keys
TPCH	8	61	8	16	6,885,051	22	8	8
AdvWork	71	486	7.5	26	754,248	33	27	45

4.4.2 Experimental Set-up

We implemented our algorithms in Java and performed experiments on an Intel Core i5 vPro 2.4GHz CPU with 8GB Ram. We first run experiments for threshold-dependent algorithms to select appropriate thresholds required for an overall comparative analysis. The range of thresholds include; NSim (0.50 - 0.95) with an interval of 0.05, Soundex (1 - 4) with an interval of 1, Sem (0.50 - 0.95) with an interval of 0.05 and Cosine (0.50 and 0.95) with an interval of 0.05. Next, we explored the performance of individual algorithms based on mean completion time over 20 runs. We then combined their predictions based on the voting scheme and hierarchical clustering. The essence is to emphasise the importance of combining individual algorithms in order to consolidate their strengths as each algorithm is limited in a particular way. Also, the experimental set-up is conducted to validate the performance of each algorithm and compare performance with the proposed ensemble methods and the state-of-the-art algorithms (FaskFK, Randomness and HoPF). Finally, we compared performance with state-of-the-art algorithms (FaskFK [76], Randomness [396] and HoPF [198]). FastFK combines heuristic features with different rules to detect foreign keys, which assumes that each table pair can hold only one foreign key. The Randomness algorithm uses a randomness metric to discover both single-column and multi-column foreign keys by using the earth-mover distance (EMD) to measure the data distribution similarity between foreign key candidates. HoPF uses score function and pruning rules for the holistic discov-

[5]http:/www.tpc.org/tpch

[6]https://github.com/Microsoft/sql-server-samples/releases/tag/adventureworks

ery of both primary and foreign keys as a subset of sets of unique column combinations and inclusion dependencies.

4.4.3 Evaluation Metrics

We employ three standard evaluation metrics to measure the performance of individual algorithms; Precision, Recall and F-Measure. Let g_1 be a graph of actual relationships between set of columns (nodes) C_1 and E_1 be the set of edges of g_1. Let g_2 be another graph containing inferred relationships between columns discovered by a discovery algorithm with a set of columns C_2 as nodes and E_2 as edges of g_2. Let $TP = E_1 \cap E_2$. TP represents true positives, a set of edges common to both E_1 and E_2 and $|TP|$ is the number of edges in TP. Let $FP \subseteq E_2 \setminus TP$ be a subset of $E_2 \setminus TP$ which represents false positives. FP and TP do not share common edges, and $|FP|$ is the number of edges in FP. Let $FN \subseteq E_1 \setminus TP$ and $|FN|$ represents the number of edges in FN. Let $x = (C \times C) \setminus \Delta$ be all edges formed from all pairs of columns, such that E_1 and E_2 are both subsets of x. Then, we define TN (True negatives) as $TN = x \setminus (E_1 \cup E_2)$ and $|TN|$ is the number of edges in TN.

Precision is computed as $\frac{|TP|}{|TP|+|FP|}$ which evaluates the percentage of relevant outcomes discovered by our algorithms. We compute recall as $\frac{|TP|}{|TP|+|FN|}$. Recall evaluates the percentage of relevant outcomes that were discovered by a discovery algorithm over the total relevant outcomes. We then compute F-measure as $\frac{2*Precision*Recall}{Precision+Recall}$ to measure the weighted harmonic mean of precision and recall.

4.4.4 Comparative Analysis

Discovery Completion Time

The mean completion time of individual algorithms is shown in Table 4.2. This involves 20 experimental runs over the TPCH dataset. The name similarity (NSim) algorithm records the lowest mean time of 4.25 milliseconds with a minimum time of 0 milliseconds and a maximum time of 16 milliseconds. On the other hand, the content-based (Col) approach takes longer than other discovery algorithms, with a mean time of 2868283.3 milliseconds (47.81 minutes). Figure 4.1 shows example of graphs generated by Sem (a) and Soundex (b) algorithms over the TPCH dataset. The example shown in Figure 4.1 gives an indication of how the individual algorithm produces different graphs based on the information available to them.

Comparison with existing techniques

We compared our results with those already published by Chen et al. [76], Zhang et al. [198] and Jiang and Naumann [396]. The specified gold standard used for evaluation is based on primary/foreign key relationship. Performance is shown in Table 4.3 - the best performance for the TPCH dataset results in an F-measure of 1.00, which Randomness achieved. The

Table 4.2: Completion Time of Discovery Algorithms in Milliseconds

Algorithms	MinTime	AveTime	MaxTime
Cosine	72	136.15	351
Pri	1228045	1501709.15	2926454
NSim	0	4.25	16
Col	2107575	2868283.3	7901629
Val	15671	16170.85	19454
Sem	3481	4491.35	8711
Soundex	3	5.95	34
Usage	144	342.2	1552

Randomness performance is largely attributed to the assumption that true primary keys exist and are known. Randomness matches the known primary keys to columns with the same names, making it possible for the algorithm to achieve that score. Our methods exploit database information differently without general assumptions about true primary key existence. Three of our methods (Sem, Usage, Cosine) outperformed the FastFK algorithm on the TPCH dataset with respective F-measure scores of 0.83, 0.80 and 0.73. The performance of the Usage-based approach is highly dependent on the quality of existing queries (i.e, views and procedures). For instance, if the queries use all the true primary keys to link tables, then an F-measure of 1.00 is possible.

Table 4.3: Comparison of Proposed Discovery Algorithms, Ensemble Strategies and state of the art results already reported in [198]

Categories	Algorithm	TPCH			Algorithm	AdvWork		
		Precision	Recall	F-Measure		Precision	Recall	F-Measure
Individual Algorithms	Cosine	0.73	0.73	0.73	Cosine	0.02	0.86	0.04
	Pri	0.24	0.91	0.38	Pri	0.03	0.90	0.06
	NSim	0.22	1.00	0.37	NSim	0.02	0.83	0.04
	Col	0.15	0.91	0.26	Col	0.01	0.93	0.03
	Val	0.08	1.00	0.15	Val	0.01	0.04	0.02
	Sem	0.77	0.91	0.83	Sem	0.02	0.83	0.04
	Soundex	0.07	0.27	0.12	Soundex	0.02	0.83	0.04
	Usage	0.89	0.72	0.80	Usage	0.14	0.58	0.23
2-Clusters Combination	Sem_Pri	1.00	0.91	0.95	Sem_Pri	0.18	0.73	0.29
	NSim_Val	0.85	1.00	0.90	Cosine_Pri	0.19	0.76	0.30
	NSim_Pri	0.91	0.91	0.91	NSim_Pri	0.18	0.73	0.29
3-Clusters Combination	Sem_NSim_Pri	1.00	0.91	0.95				
	Sem_NSim_Col	1.00	0.82	0.90				
	Sem_NSim_Val	0.91	0.91	0.91				
Voting	PVote50	0.79	1.00	0.88	PVote75	0.18	0.80	0.29
	PVote62.5	0.85	1.00	0.92	PVote87.5	0.16	0.49	0.25
	PVote75	1.00	0.91	0.95				
State-of-the-Art	FastFK	0.56	0.90	0.69	FastFK	0.32	0.97	0.49
	Randomness	1.00	1.00	1.00	Randomness	0.90	0.41	0.56
	HoPF	0.88	0.88	0.88	HoPF	0.31	0.84	0.46

Regarding the AdvWork dataset in Table 4.3, our algorithms could not achieve significant F-measure results apart from the Usage-based algorithm that achieved an F-measure score of 0.23. The poor performance is largely attributed to a huge number of false positives discovered by our methods. These false positives are caused by the inherent semantic relationships not defined in the primary/foreign key relationship we have used as the gold standard in our evaluation. Regarding recall, Content-based (Col) and Primary key (Pri) al-

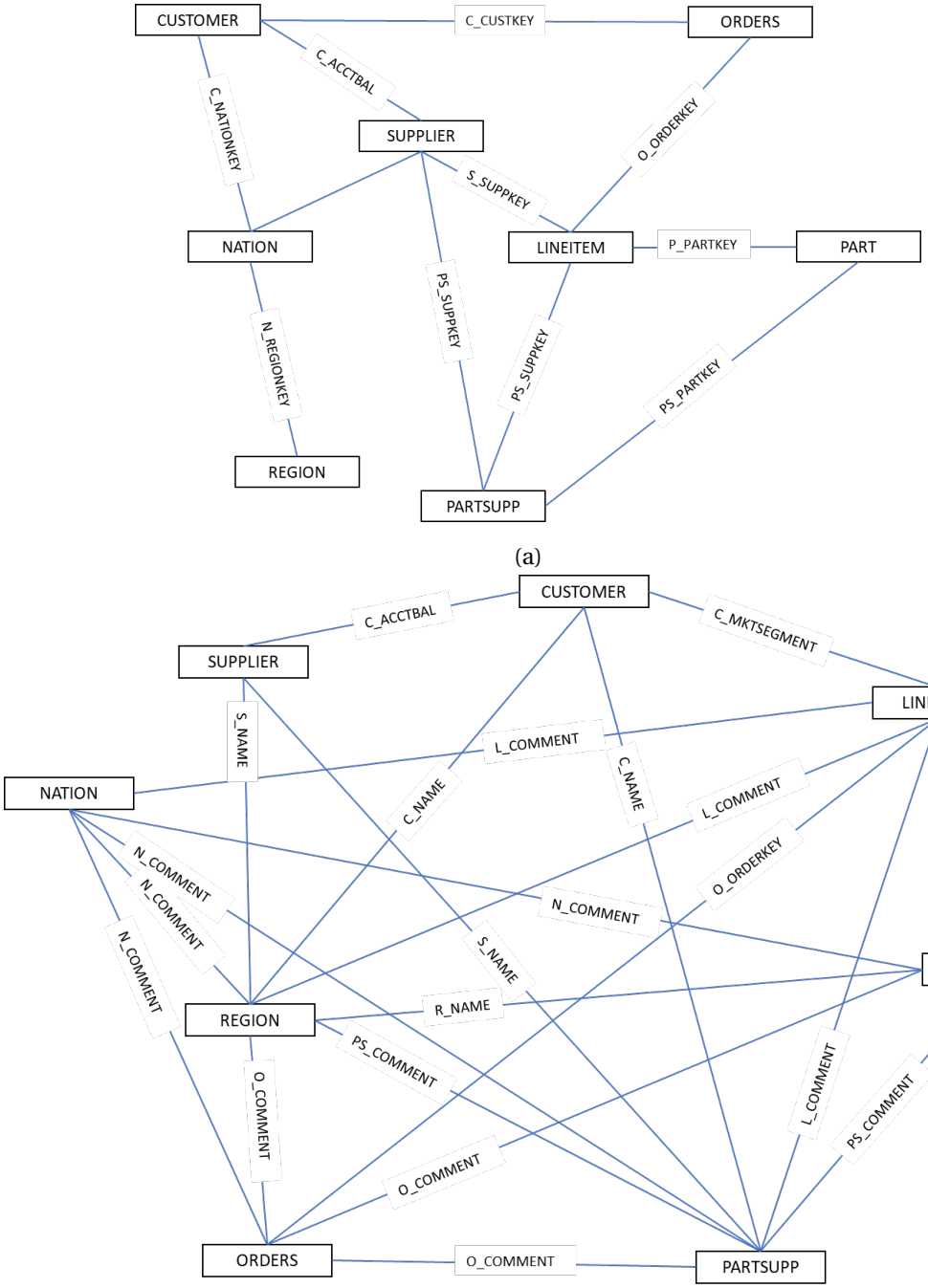


Figure 4.1: Graph Example

gorithms achieved 0.93 and 0.90, respectively.

Overall, the diversity displayed by the individual algorithms is based on the characteristics of the data. The algorithms have performed in different ways over the two datasets. However, the diversity of the independent algorithms can be exploited by combining their outcomes in different ways to improve performance.

Ensemble Performance

We used voting strategy and hierarchical clustering to combine the diversity of the outcomes produced by the individual algorithms and compare performance with the state-of-the-art algorithms reported in [198]. The results achieved by the proposed ensemble strategy is highlighted in Table 4.3. Voting thresholds are given as;
$P_{weighting} = (12.5\%, 25\%, 37.5\%, 50\%, 62.5\%, 75\%, 87.5\%, 100\%)$. We include results of the top three voting thresholds over the two datasets (TPCH and AdvWork) in Table 4.3. For TPCH dataset, the three top voting thresholds, 75%, 62.5% and 50% (i.e., PVote75, PVote62.5 and PVote50) achieve respective f-measure scores 0.95, 0.92 and 0.88, precision scores 1.00, .85 and 0.79 and recall scores 0.91, 1.00 and 1.00. The voting scheme could not reach the F-measure score (1.00) delivered by the Randomness algorithm. However, a 0.95 score was achieved, which outperformed HoPF and FastFK.

In the AdvWork dataset, despite the poor performance of the individual approaches, the voting scheme helped improve the performance. Although, this strategy could not outperform the selected state-of-the-art algorithms. The reason for this is due to the existence of several semantic relationships which are not explicitly specified in the database structure. We only relied on the explicit specifications of primary key/foreign key relationships for our evaluation.

The drawback of the voting strategy is that the voting strategy considers all the discovery algorithms. This is quite expensive in terms of computational time. For instance, based on Table 4.2, the total average completion time to implement a voting strategy will take about 4391143.2 milliseconds (73.19 minutes). However, this could be addressed by using an appropriate sophisticated parallel computing approach, which is beyond the scope of this thesis.

In terms of the hierarchical clustering strategy, with the TPCH dataset, we evaluate two clusters and three cluster combinations. We obtain 15 unique combinations of algorithms with two clusters and 18 unique combinations with three clusters. For instance, the best performance in the two clusters combination is produced by Sem_Pri. Sem_Pri gives an f-measure score of 0.95 with a precision score equal to 1.00. This prediction means that no false positives were predicted with the combined efforts of both Sem and Pri algorithms. Similarly, Sem_NSim_Pri obtains an f-measure score of 0.95 with a precision score of 1.00. When comparing performance with state-of-the-art algorithms, Sem_Pri and Sem_Nsim_Pri give better performance than HoPF and FastFK.

In the AdvWork dataset, two clusters were predicted by the clustering algorithm. We obtained the top three unique combinations of two clusters based on f-measure performance. The best performance is produced by combining Cosine and Pri with an f-measure score of 0.30, precision score of 0.19 and recall score of 0.76. The results reported by the state-of-the-art algorithms outperformed the combined efforts of Cosine and Pri. See Table 4.3. We have attributed the poor performance over the AdvWork dataset to the lack of sufficient

gold standard used in the study. We only relied on the primary/foreign key relationships specified in the database. An expert opinion would be needed for additional information about the semantic relationship.

Overall, results show clearly that some specific algorithms are relevant when combined in certain ways. The Pri, for instance, tends to perform well when combined with algorithms like Sem, Nsim or Cosine, irrespective of the data characteristics. However, the suitability of Pri is impaired due to speed considerations. Therefore, combining algorithms depends largely on the user's compromise on speed, reliability and sufficiency.

4.5 Summary

This chapter relates to objective (O3) and addresses the research question (RQ3) involving fragmented supply chain data linkages. This chapter relates to the data layer in the methodological framework presented in Section 5.3 of Chapter 5. The chapter investigated eight discovery algorithms and showed how their predictions could be combined to identify more comprehensive links between supply chain data. The discovery algorithms identify potential links in different ways based on different levels of database information. In evaluating the performance of our approaches, based on two diverse datasets, we showed that different levels of schema information could be exploited and combined to reduce the generalisation error associated with each algorithm. We showed in our experiment that an appropriate combination strategy could be adopted to improve relationship discovery outcomes. The performance of individual discovery algorithms is limited, indicating the necessity to combine several algorithms to bring together their strengths. The performance of our algorithms is compared with state-of-the-art algorithms using precision, recall and f-measure performance metrics. The next chapter considers the properties of the linked components in the supply chain from data and use these properties to define and develop a formulation to support the holistic optimisation of the problem.

5

Linked Problem Systems: Methodological and Implementation Framework

5.1 Overview

This chapter attempts to answer one of the critical questions raised in the previous chapter, i.e., "how can we use a framework to state what the holistic linked optimisation problem is?". More specifically, the chapter explores a methodological framework for Linked Problem Systems (LPS) and further presents several modes of linkages using a set of supply chain decision problems. This chapter further presents insights into understanding linked problems as a networked system.

The rest of this chapter is organised as follows: Section5.2 explores different interactions in LPS and their behaviour. In Section 5.3, the chapter presents a methodological framework for LPS from a supply chain perspective. Section 5.4 explores several modes of linkages using a set of supply chain decision problems. The section also presents the LPS implementation framework that allows the formulation and solution of LPS benchmarks. Section 5.5 provides an exploratory, experimental analysis the linked problems formulated in Section 5.4 to ascertain the interactions between the selected problem combinations. Lastly, Section 5.6 provides a summary of this chapter.

5.2 Complex Interactions

The concept of supply chain has attracted different perspectives in academia and practice. Many consider supply chain in operational terms, while others in management philosophy or process [332]. From Yates [384] perspective, the system of supply chain is complex, and as such, the following features are evident according to [332]: various actors; high number and several relations, processes and actors interactions; dynamic processes and interactions; system involves many process levels; and a large amount of information required for system control.

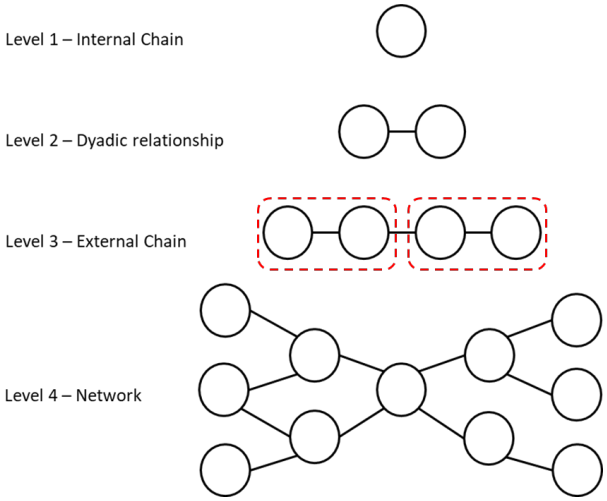


Figure 5.1: Supply Chain System Levels [165]

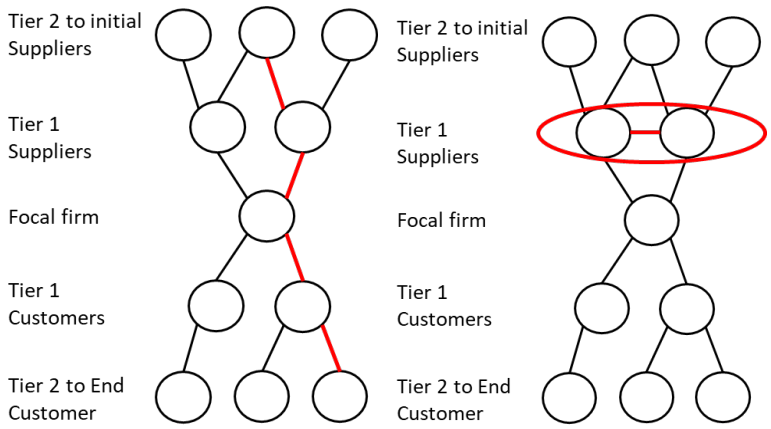
The concept of systems thinking in supply chains is a way to understand its complexity holistically. In other words, the system approach to supply chains can provide a holistic framework for understanding various complex interactions that exist. The system embodies the idea of a set of connected elements which form a whole [18]. A different aspect of a system can be distinguished according to [331]; functional, structural and hierarchical.

In a functional system, a system behaves independently of its realisation. Structural system behaviour is understood from the relations of a set of interlinked elements which form the whole system. The hierarchical system considers parts of the system as sub-systems, and the system itself as a more comprehensive system [331].

Harland [165] argues that, from a system approach viewpoint, the conceptual development of supply chain occurred at the different levels of relationships, chains and networks. According to Figure 5.1, these levels can be explored to examine supply chain interactions. In system level 1, interactions occur within the internal chain of the organisation. System level 2 represents a dyadic or two relations containing inter-organisational interactions. System level 3 is an external chain of a set of dyadic relations. In system level 4, the interactions involve a network of interconnected chains.

The interactions in a supply chain are considered from two viewpoints; vertical and horizontal/parallel. Vertical/sequential interaction involves several inter-organisational interactions of actors at different tiers/echelons. The horizontal interaction consists of interactions within the same tier in which the actors in the tier play the same role in the supply chain. Horizontal interactions exist between competitors. Different factors drive the incentives for horizontal interactions. For instance, two competitors may decide to split the workload or collaborate to fulfil a dominant customer's overwhelming orders [18].

Actors in a supply chain interact in pursuit of expected benefits involving complementary resources, skills, and capabilities. However, the imbalanced distribution of such shared benefits might require re-balancing the supply chain. This creates complex interactions as



(a) Vertical Interaction (b) Horizontal Interaction

Figure 5.2: Forms of Interactions [18]

cooperation will mean that all actors collaborate to create value mutually. Also, at the same time, individual interest sets in and actors would have to gain such value by bargaining [408]. Therefore, the interaction could be considered from three behavioural perspectives; transaction, collaboration, and integration [18]. We characterise transaction by competitive behaviour, while collaborative behaviour is driven by cooperation. Integration explains the behaviour that underlies the coexistence between competition and collaboration. The behavioural situation is evident in a supply chain, and therefore, as such situations continue to become more uncertain and ambiguous, this results in conflicts and opportunistic behaviour [408].

5.2.1 Network Interactions in Supply Chain

Network interactions are understood from network theory which provides the origins and characteristics of networks formed from components of various complex systems. Several studies have used network science to solve complex system. A network interaction provides a holistic view by considering the topological characteristics of the network. Therefore, studying supply chain networks requires a new understanding of their underlying structure, properties, and interaction types [229]. From an evolutionary point of view, according to Daoutidis et al. [98], the natural philosophy behind the different network structures can be understood from their evolutionary origin, which can be adapted to improve our comprehension of real-world networks or linked structures. Hence, there is motivation to leverage supply chain network structures for the algorithmic design of complex optimisation problems [98]. In investigating network interactions in the supply chain of a wine industry, Saglietto et al. in [325] used a social networks analysis to represent its structure and presents the structure as an essential tool for analysing the performance of the underlying supply chain. István and Tamás, in [189], applied network science theory to determine the cause and the effect and behaviours of the relationships between network elements in a logistic network.

This has motivated this research into abstracting LPS from supply chain network structures and creating a framework in which their properties can be studied.

Over the past ten years, supply chain has grown longer and become interconnected due to globalisation, and the rising cost pressures [82]. According to [230], from the network science framework, the studies of network topology aim to present interconnected systems in terms of their behavioural phenomena.

In the work of Ledwoch et al. [230], two characteristically distinct network topologies were considered; random networks and scale-free networks. Random networks are networks with Poisson degree distribution where links between components of the networks are placed at random. The random networks serve as benchmarks for verifying the existence of certain features of a topology. Scale-free networks are networks with a power-law degree distribution. They are made up of significant hub components with many links and minor components connected to these hubs [230].

Chin and Lee establish a broad set of topological characteristics in real supply chain networks, which describes how to build the topology and robustness of supply chain networks in complex networks [77] [200]. Brintrup et al. [48] believe that communities connected by hub firms form the structure of an industry. Their empirical study has shown in the literature that supply chain networks follow a scale-free pattern [48].

5.3 Methodological Framework

Supply chains are increasingly complex and information-driven. Different units in the supply chain make decisions that have consequences for successors in the chain. We are interested in supply and service chains that can be modelled as linked optimisation problems, i.e., Unit A selects a solution to an optimisation problem, and that solution has the consequence of creating an instance of an optimisation problem for Unit B, its successor in the chain. Our approach is to create an analytical and programmatic framework for linked optimisation problems so that supply chains can be studied systematically. Our methodological framework is based on three analytical layers as presented in Figure 5.3 - data layer, supply chain process model layer, and supply chain optimisation layer.

This approach is highly data-driven, and as such, it entails using documents and records as one of the data collection techniques in gathering primary data, which consists of existing data in the form of databases and reports. We relied on benchmark problem instances for the study to test our model. This is because benchmark problem instances are already academically well-known problem sets around which efficient algorithms have been developed. They are believed to offer a broad range of challenges to optimisation algorithms. They are all motivated by real-world operational management problems and are frequently adapted to practical applications. Artificially (randomly) generated problems are not generally thought to be realistic, so the benchmarks are preferred. There is a challenge in **how** to link them, and that is what this thesis is addressing.

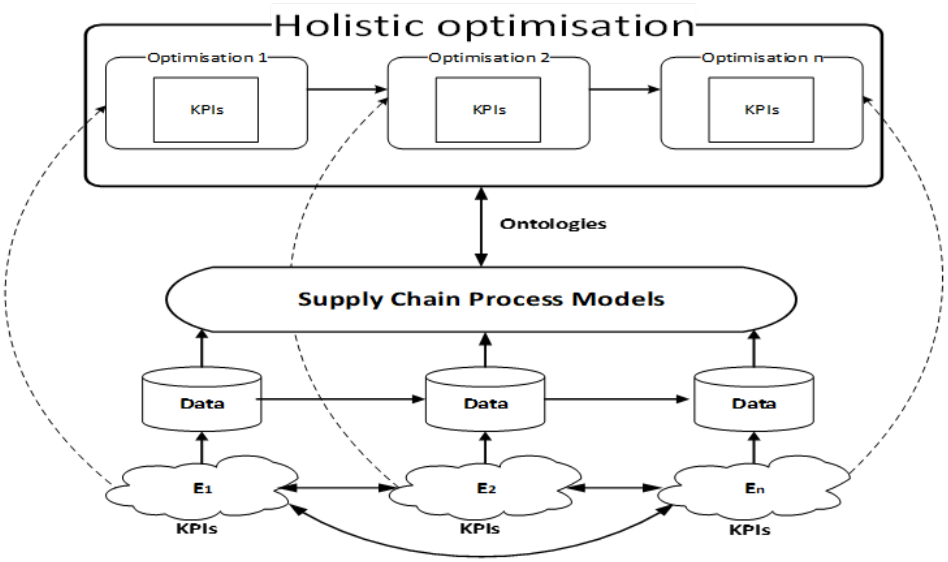


Figure 5.3: Methodological Framework for Linked Optimisation Problem

An immense corpus of data was accessed from a multinational telecommunication company during this research. This data allows the researcher to gain knowledge of the real-world supply chain as data is entirely real and contains daily business transactions from different functional units of the company. Specifically, the methodological framework and the data-linking algorithms developed in this research were inspired by the company's data due to the complexity and lack of coherence observed in the data. We observed many collaborators, processes, and technologies in terms of complexity. This accounts for complex decision-making processes in the different functional units of the organisation. Furthermore, each functional unit maintains an extensive database, which is differently configured from other functional units. This accounts for the lack of data coherency across the organisation. Hence, decisions are made in isolation.

Three significant limitations were encountered in the course of exploring the data of the telecommunication company. There is restricted use of the data as data contains sensitive business information. Because this information needs to be kept private, benchmark datasets were used to test the performance of the algorithms developed. The benchmark dataset contains similar business activities and schematic information of typical supply chain data. Second, the observed company's data is enormous and lacks integrity as data is obtained from heterogeneous sources. These sources form the different functional units of the organisation where data is propagated. Specifically, data comprises millions of records, resulting in high time complexity during an algorithmic (data-linking) search. Different levels of information in the datasets were exploited to address the time complexity issues. Lastly, the company's data lacks documentation. There is a lack of understanding of the comprehensive information and a lack of different timestamps to understand the sequence of events. To gain such understanding would require highly time-intensive human efforts,

which might be error-prone and costly.

5.3.1 Data Layer

Millions of records of data are generated from collaboration processes among the components of a supply chain. These pieces of data contain event information which is stored in unstructured form. The extraction of such data forms a significant aspect of the optimisation of linked problems because problem linkages are evident in the data they generate due to the processes executed at different levels of the problem. In most cases, extracting them requires effort as they are highly fragmented over many database sources. It is, therefore, essential to link these pieces of fragmented data in a logical way to discover the dependencies between the components of the supply chain.

As seen in Figure 5.3, the data layer from the supply chain domain manages the integration of fragmented data from different components of a supply chain system. Fragmented data is propagated due to business interactions between supply chain participants. The interactions involve several processes, which result in hundreds or thousands of tables and thousands of attributes in a database.

The data layer aims to identify ways to capture and visualise relationships, which is a prime prerequisite to understanding links between billions of data records generated to extract insights for commercial advantages. This layer investigates how relationships may be inferred from a database's schematic and semantic data representation.

5.3.2 Supply Chain Process Model Layer

The layer attempts to mine and learn from an analytical perspective supply chain processes that run across fragmented parts of the supply chain. This layer combines data relationships into process pathways based on process mining techniques to obtain process models depicting a supply chain process sequence. The data layer attempts to transform fragmented data into a structured data format containing semantically meaningful attributes. This layer uses structured data to extract business events. It mines and learns what event type, time stamps and a set of resources/attributes are associated with a business event. The process may involve linking several data columns to a single or several business events. Techniques deployed at this layer use a set of graph-based discovery algorithms to determine how certain events are related and how such events result in a set of process instances.

1. Input Parameters
    a) Related entities/concepts extracted from a graph
    b) A set of event data.
    c) A suitable process mining and learning algorithm.
2. Output Parameters

a) Business process descriptions derived from the event data.

b) A global graph containing a sequence of processes.

This layer defines the level of dependencies among the functional units where different optimisation problems emanate. A representation of the dependencies is generated as a directed graph. The supply chain optimisation layer relies on this layer to define the linkages in the holistic optimisation process. However, due to data confidentiality issues and the lack of coherent event data, we relied on human judgement on the perceived dependencies among the components of the linked problem. Therefore, the process model layer was not explored in the study. Instead, we assumed an initial process analysis that already suggests the process sequence of the optimisation problems.

5.3.3 Supply Chain Optimisation Layer

Optimisation is an effective tool that automatically identifies optimal result from a set of possible options. Optimisation entails three significant processes; optimisation model, techniques and analyses of optimisation results. The optimisation layer uses the understanding of the dependencies to design appropriate optimisation techniques. In the methodological framework, this layer requires information about the process pathway to guide the sequence of optimising the problems in the linked structure.

From Chapter 3, we represent a formalism of linked optimisation problem as P, containing n connected problems. P is defined as $P = \{p_1, p_2, \cdots, p_n, (D)\}$ where D is a representation of the linkages between the problems in P. D describes the connectedness, a directed graph obtained from the supply chain process model layer.

This layer can adopt at least two approaches to solving the linked problems.

- Concatenation Approach
- Sequential Optimisation Approach

The first approach requires combining solutions (i.e., concatenation) as a joint or holistic solution where the joint solution is evaluated using a holistic fitness function or as a multi-objective function. This approach's challenge is identifying the best way to combine them, considering that the different solutions might have utterly different solution spaces. If, for instance, we aimed at solving the linked problem using an evolutionary algorithm, like GA, the problem representation is critical; therefore, more investigation would be needed. In addition, the designation of operators for the specific solution representation also poses a potential problem. A typical way this can be viewed is to consider the problem as a multi-objective optimisation. The multi-objective approach requires re-modelling the problem as a joint or a single solution-based problem containing multiple optimisation goals. However, the presence of different solution spaces introduces several search algorithmic design

options for such problems [102]. The second approach is to consider the problem as a sequential optimisation approach. The approach induces a hierarchy between two or more related problems. However, this hierarchy might translate to a higher computational complexity than considering the sub-problems simultaneously [232]. We adopted two algorithmic methods: Sequential algorithmic design and Concatenation algorithmic design from the perspective of the optimisation layer in our framework.

From this layer, the study investigates two case studies in Chapter 6, linked optimisation of a distributed manufacturing system involving two classical problems - Facility Location Problem (FLP) & Permutation Flow Shop Scheduling Problem (PFSP), and a service chain problem involving Job Assignment Problem (JAP) & Traveling Salesman Problem (TSP).

Sequential Algorithmic Design

A sequential approach uses the optimal solution of the root problem and feeds it into the following problem as input to produce a solution. The sequential approach uses a hierarchical process. The output of the root problem is injected as input for a lower problem in the hierarchy until the last problem (leaf node) is solved. In a sequential approach, an algorithm is assigned to each problem in the linked structure.

In a real-world supply chain, each problem has a manager in decision-making. An implication is that the individual managers might seek to optimise their operations by thinking about their supply chain component without much interest in sharing information with others in the supply chain network. In our implementation described in Section 5.4.3, such an issue can be addressed from our implementation framework when embedded in the sequential process, provided that the knowledge of the process pathway involving the components of the supply chain is available. With the given information, a solution obtained from a principal problem can be used to instantiate a successor problem, as shown in Figure 5.4.

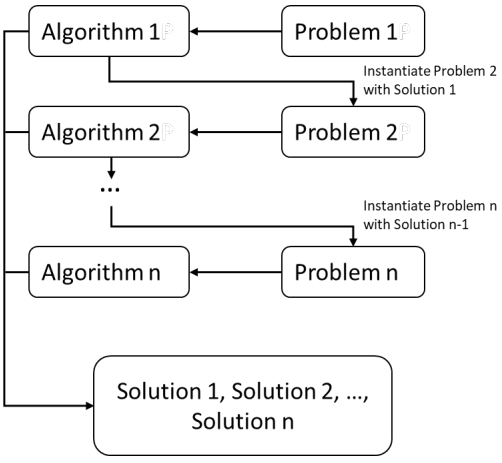


Figure 5.4: Sequential Approach

Concatenation Algorithmic Design

The concatenation approach embeds n problems in a given algorithm while simultaneously considering the different configurations of the problems in the linked structure. The approach uses specially designed operators to enhance the algorithm performance further to explore complex search spaces relating to the linked problem [250]. The exploitation and exploration process embedded in the search process of the concatenation approach considers the individual problem characteristics. Two of the methodologies utilised in Chapter 6 embed a multi-objective approach and multi-criteria decision-making method to select the best solution pairs in the linked problems considered.

Figure 5.5 depicts a generic concatenation methodology. The algorithm takes in all the problems and embeds generic solution representations. The algorithm also embeds a linked implementation concept that allows the interaction between the solutions of the related problems. During the interaction, a solution to the principal problem is fed to instantiate a dependent or set of dependent problems. Different multi-objective of multi-criteria decision-making methods can be adopted as selection operators as these consider the solution spaces of all the problems without losing the generality of the linkages among the problems. In addition, other forms of operators, peculiar to each solution space, can be embedded in the algorithm to improve the different solution mixes. A typical concatenation approach is seen in [182]. A cooperative coevolutionary approach was proposed to tackle a supply chain problem involving Job Shop Scheduling Problem (JSSP) & Vehicle Routing Problem (VRP). The approach embeds two algorithms which are run in parallel. Each algorithm corresponds to each problem, and the cooperation between the two problems occurs while evaluating individual solutions.

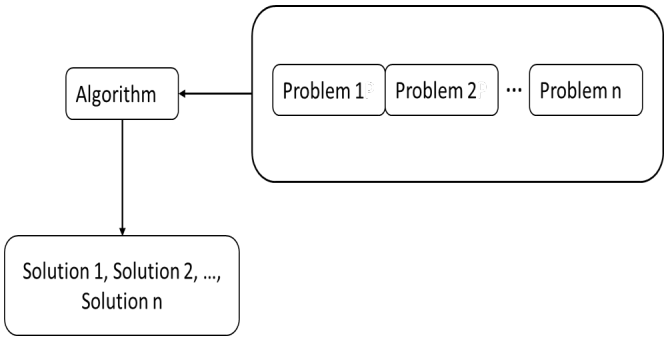


Figure 5.5: Concatenation Approach

Figure 5.6 better explains the typical problem scenario and how the framework can be adapted to this problem type. Assuming that, according to Figure 5.6, fulfilling customers' orders has to involve three functional units of a typical telecommunication company - customer care, warehouse, and engineers. Customer care is responsible for taking orders and making certain decisions to minimise order completion time. Customer care informs the warehouses to request items to fulfil individual customers' orders. The warehouses prepare

the items for pick-up. Likewise, the engineers are alerted for tasks and to visit the warehouses to pick up the items. In a typical telecommunication organisation, orders could involve installations and repairs at the customer's location. Field engineers perform these tasks at different customers' locations. The engineers must visit the warehouses to pick items required for the tasks to be performed at the customers' locations. The engineers are constrained by time, which means they have to make decisions to minimise, for instance, their travel time and waiting time. To reduce travel time, the engineers must pick up all the required items at once for all the customers they visit. Picking all the required items at once prevents the engineers from visiting the warehouses multiple times. It means that at the warehouses, items for the customers have to be available for pick-up. However, it might not be the case due to the cost of holding the items. The warehouse also wants to minimise the stock level so that the holding cost of items is reduced. Minimising stock level has an impact on the engineer's waiting time as well as the overall completion time of tasks. This is because items are only made available whenever they are needed. This results in conflicting objectives between the warehouse and engineers' units.

The framework presented in Figure 5.3 can be used to manage the linked problem shown in Figure 5.6. The framework enables a process of automation and optimisation, which can help to maximise the operational effectiveness of the task completion process. At the data layer, each functional unit of the chain generates data that reflects its operational activities. The data layer automatically infers from the data generated from the activities of these functional units, using a suitable approach to form a connection between them. The information about customers' jobs is linked to the resources required to fulfil the jobs. Likewise, the job information is linked to the engineers' data as individual jobs may require a different skill-set. Next, the process model layer uses the data linkages and event logs to form a sequence of activities. The process layer captures the overall processes. The overall sequence of processes in the order fulfilment forms a directed graph D. The graph D is used in the optimisation layer to describe the linkages shown by the arrows in Figure 5.6 between the functional units. The optimisation layer applies the knowledge of the linkages to design an appropriate algorithmic approach that will optimise the overall process. Several approaches can be adopted to optimise the linked problem. However, each approach comes with its trade-off implications. For instance, some algorithmic approach might provide a balanced compromise of objectives between the functions, while some might trade off the stock level to speed up the completion process.

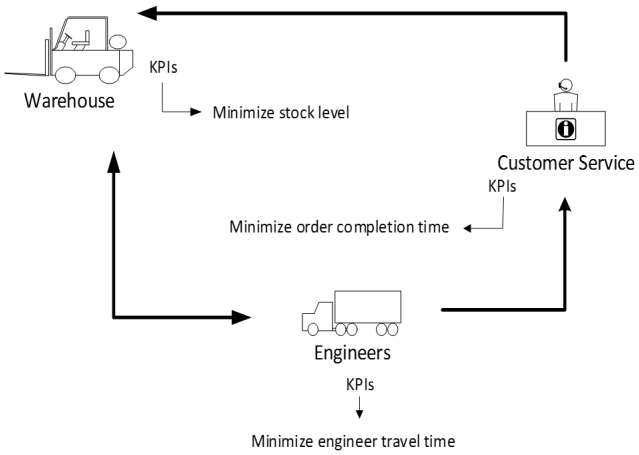


Figure 5.6: Typical Example of Linked Optimisation Problem

Next in this chapter is to explore several modes of linkages using a set of supply chain decision problems. This includes a representation of the LPS implementation framework that allows the formulation and solution of LPS benchmarks in order to gain insights into understanding linked problems as a networked system.

5.4 Linked Problem Framework

Unsurprisingly, linked problem patterns can be recognised in the literature. In Chapter 3, this study extracted the concept of a linked problem from the literature and came up with a formalism to describe it. However, its conceptual framework has not been developed. In light of future directions, we must develop a framework to systematically study the linked problem as a formal topic in an optimisation problem. The need for a formal framework is to seek holistic treatment for obtaining an optimal solution, such that the underlying dependencies among the sub-problems are considered. A linked problem is a real-world problem that often requires such treatment, so it is not easily decomposable; hence, the individual sub-problems cannot be solved in isolation [41]. In addition, the theoretical investigation of computational intelligence methods for linked problems poses a new challenge, requiring an appropriate framework. To the best of our knowledge, there is lack of complex computational analysis to provide insights into the interactions between the different sub-problems [41]. Therefore, a formal, conceptual framework will be beneficial.

5.4.1 Problem Formulations

As a starting point, we use a mix of classical optimisation problems peculiar to supply chain decision problems to describe different modes of linkages in linked optimisation problems. The classical problems include; Facility Location Problem (FLP), Knapsack Problem (KP),

Job Assignment Problem (JAP), Permutation Flow Shop Scheduling Problem (PFSP), Travelling Salesman Problem (TSP), and Quadratic Assignment Problem (QAP).

Facility Location Problem (FLP)

A matrix of size $J \times K$ defines cost d_{kj} to fulfil customer j demand by facility k. Let D_j be the units of customer j demands and w_k be the capacity of facility k. The assignment of customer $j \in J$ to a facility k is depicted by y_{kj} where $y_{kj} \in \{0,1\}$. A fixed cost α_k is incurred when a facility is selected to fulfil a customer's demand. FLP seeks $x_{_{FLP}} = \{x_1, \cdots, x_K\} \in \{0,1\}^K$ that minimises;

$$minf(x_{_{FLP}}) = \sum_{k=1}^{K} \alpha_k x_k + \sum_{k=1}^{K} \sum_{j=1}^{J} d_{kj} x_k y_{kj} \tag{5.1}$$

Subject to;

$$\sum_{k=1}^{K} \sum_{j=1}^{J} y_{kj} = 1 \tag{5.2}$$

$$\sum_{j=1}^{l} D_j \leq w_k \quad \forall k \tag{5.3}$$

$$x_k, y_{kj} \in \{0,1\} \quad \forall j,k \tag{5.4}$$

Constraint 5.2 ensures that each customer demand is fulfilled by one facility. Constraint 5.3 ensures that the total demand of customers assigned to each facility does not exceed the capacity of the facility. Constraint 5.4 defines the decision variables. An instance of FLP is recorded in Table 5.1. The fixed cost of the facility, customer cost per facility and unit of demand per customer are recorded in Table 5.1. So, for instance, fulfilling job C_1 at facility 1 will cost 20 (the fixed cost) + 34 (the total demand cost).

Table 5.1: FLP Example - 5 Facilities and 8 Customers

		Demand Cost to Facility				
Customers	Unit of Demand	F1	F2	F3	F4	F5
C1	5	34	20	76	54	44
C2	17	34	68	23	27	43
C3	3	56	20	65	53	29
C4	8	68	25	47	42	60
C5	16	45	33	65	10	36
C6	10	51	30	71	60	87
C7	9	30	35	81	50	65
C8	6	25	90	21	30	42

Facilities	Fixed Cost	Facility Capacity
F1	20	25
F2	23	30
F3	25	26
F4	29	19
F5	30	20

Knapsack Problem (KP)

Given a set of n potential items $I = \{I_1, I_2, \cdots, I_n\}$, a value ρ_i of each item to be picked, weight w_i of each item and a capacity W for the knapsack. We seek to find the selection of items $x_{KP} = \{x_1, \cdots, x_n\} \in \{0,1\}^n$ that maximises;

$$maxf(x_{KP}) = \sum_{i=1}^{n} \rho_i x_i \tag{5.5}$$

Subject to;

$$\sum_{i=1}^{n} w_i \leq W \tag{5.6}$$

$$x_i \in \{0,1\} \quad \forall i \tag{5.7}$$

Constraint 5.6 ensures that the total weights of selected items must not exceed the capacity of the knapsack. Constraint 5.7 defines the decision variables.

Table 5.2: KP Example - 6 Items

	I_1	I_2	I_3	I_4	I_5	I_6	I_7	I_8
Item Value	39	40	35	60	79	27	50	35
Item Weight	6	7	5	12	15	3	10	7
Knapsack capacity	40							

Job Assignment Problem (JAP)

Given a set of n jobs J such that, $j \in J, j = 1,2,\cdots,n$. Let I be the set of m agents, such that, $i \in I, i = 1,2,\cdots,m$. An $n \times m$ cost matrix defines the cost c_{ij} for assigning job j to agent i. An $n \times m$ matrix defining the resource r_{ij} required by agent i to perform job j. A capacity b_i for each agent i. We seek to find the assignment of jobs to agents $x_{JAP} = (\mathrm{x}_1, \mathrm{x}_2, \cdots, \mathrm{x}_\mathrm{n})$, such that, $j = 0,1,\cdots,n$ and $\mathrm{x}_j \in \{1,2,\cdots,m\}$, that minimises the costs associated with assigning agents to fulfil the jobs. We defined y_{ij} as a binary variable that indicates 1 if an agent i is assigned to fulfil job j and 0 otherwise.

$$minf(x_{JAP}) = \sum_{i=1}^{m} \sum_{j=1}^{\mathbf{n}} c_{ij} y_{ij} \tag{5.8}$$

Subject to:

$$\sum_{i=1}^{m} y_{ij} = 1 \quad \forall j = 1,2,\cdots,n \tag{5.9}$$

$$\sum_{j=1}^{\mathbf{n}} r_{ij} y_{ij} \leq b_i \quad \forall i = 1,2,\cdots,m \tag{5.10}$$

$$y_{ij} \in \{0,1\} \quad \forall i,j \tag{5.11}$$

Constraints 5.9 ensures that each job is assigned to precisely one agent, Constraints 5.10 ensures that the total resource requirement of jobs assigned to each agent does not exceed the agent's capacity and Constraints 5.11 defines the decision variables. Table 5.3 shows an example of JAP problem instance containing 5 agents and 8 jobs. The values in the table are the costs of assigning each job to agents.

Table 5.3: JAP Example - The assignment of 5 Agents to perform 8 Jobs

Jobs/Agents	A	B	C	D	E
J1	23	22	20	30	15
J2	12	10	15	20	25
J3	20	22	15	21	23
J4	4	6	7	5	8
J5	60	70	55	77	50
J6	30	26	38	29	34
J7	3	7	4	2	1
J8	9	5	5	9	8

Permutation Flow Shop Scheduling Problem (PFSP)

Given a set of n jobs $\{x_i\}_{i=1}^{n}$, a set of m machines $\{y_j\}_{j=1}^{m}$. PFSP is defined by $n \times m$ matrix of processing times involving n jobs processed on a set of m machines. We seek to find the permutation of n jobs $\mathbf{x}_{PFSP} = (\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n)$ that minimises the makespan;

$$minf(x_{PFSP}) = C_{max} \tag{5.12}$$

Subject to;

$$C_{1,\mathbf{x}_1} = \rho_{1,\mathbf{x}_1} \tag{5.13}$$

$$C_{1,\mathbf{x}_j} = C_{1,\mathbf{x}_{j-1}} + \rho_{1,\mathbf{x}_j} \quad \forall j > 1 \tag{5.14}$$

$$C_{i,\mathbf{x}_1} \geq C_{i-1,\mathbf{x}_1} + \rho_{i,\mathbf{x}_1} \quad \forall i > 1 \tag{5.15}$$

$$C_{i,\mathbf{x}_j} = max\left\{C_{i,\mathbf{x}_{j-1}}, C_{i-1,\mathbf{x}_j} + \rho_{i,\mathbf{x}_j}\right\} \forall j > 1, i > 1 \tag{5.16}$$

$$C_{max}(\mathbf{x}_{PFSP}) = C_{m,\mathbf{x}_n} \tag{5.17}$$

C_{max} denotes the makespan minimisation as the optimisation criterion of PFSP. Constraint 5.13 ensures that the processing of the job on each machine only starts when the processing of the same job on the previous machine is completed. Constraint 5.14 ensures that each job can start only after the previous job assigned to the same machine at the same factory has been completed. Constraint 5.15 formulates the makespan and constraints 5.16.

Table 5.4: PFSP Example - 5 Machine and 8 Jobs

	Processing Time				
Jobs	Machine 1	Machine 2	Machine 3	Machine 4	Machine 5
C1	2	3	1	2	3
C2	1	2	1	4	1
C3	2	1	2	4	3
C4	3	4	2	4	2
C5	1	3	3	2	3
C6	4	2	4	1	2
C7	1	3	1	4	1
C8	2	3	1	2	4

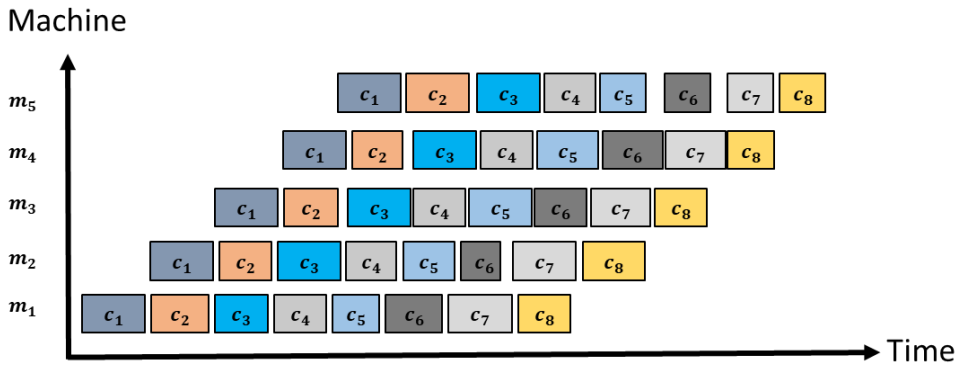


Figure 5.7: Example - Scheduling Plan

Travelling Salesman Problem (TSP)

TSP is one of the famous classical optimisation problems that involves determining a tour that minimises the total distance travelled by a salesman [151]. TSP is defined by $n \times n$ distance matrix of n cities where the salesman is required to visit each city once. The distance between two locations i and j is defined by d_{ij}. TSP seeks to determine the best permutation $\mathbf{x}_{TSP} = (\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n)$ that minimises $f(\mathbf{x}_{TSP})$ defined by:

$$minf(\mathbf{x}_{TSP}) = \sum_{j=2}^{n} d_{j-1,j} + d_{n,1} \tag{5.18}$$

Quadratic Assignment Problem (QAP)

QAP is a generalisation of a linear assignment problem with a non-linear objective function [197]. Given a set of n facilities and n locations, QAP seeks to find an optimal assignment $\mathbf{x}_{QAP} = [\alpha_{ij}]n \times n$, a permutation matrix, that minimises $f(\mathbf{x}_{QAP})$, the total operation cost of the facilities. We specify a cost c_{ij} which is associated with the cost of assigning facility i to

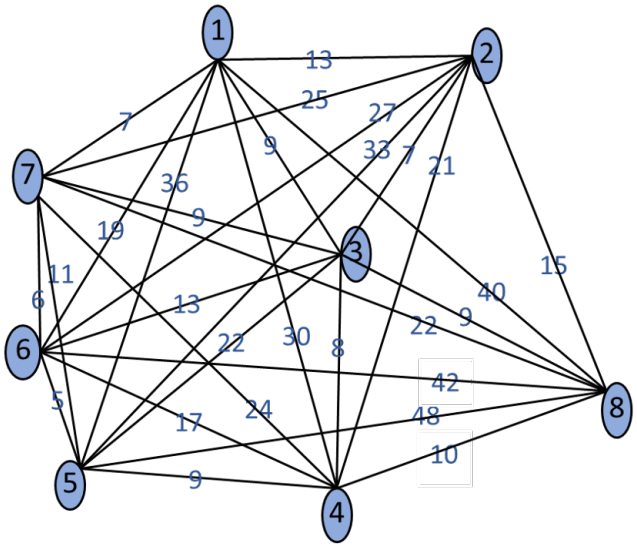


Figure 5.8: Example - TSP

location j. We define $\alpha_{ij} \in \{0,1\}$ where $i, j = 1,2,\cdots,n$ as a binary variable and if α_{ij} indicates 1, it means that facility i is assigned to location j and 0 otherwise.

$$minf(\mathbf{x}_{QAP}) = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}\alpha_{ij} \tag{5.19}$$

Subject to;

$$\sum_{i=1}^{\mathbf{n}} \alpha_{ij} = 1 \quad \forall j = 1,2,\cdots,m \tag{5.20}$$

$$\sum_{j=1}^{\mathbf{n}} \alpha_{ij} = 1 \quad \forall i = 1,2,\cdots,m \tag{5.21}$$

$$\alpha_{ij} \in \{0,1\} \quad \forall i, j \tag{5.22}$$

Constraint 5.20 ensures that each location must only be placed by one facility. Constraint 5.21 ensures that each facility must be assigned to one location only. Constraint 5.22 defines the decision variable.

5.4.2 Linked Problems and Formulations

Links in real-world applications are exhaustive. Several of these links have been selected to explore different modes of linkages in linked optimisation problems. In this section, we provide a mix of optimisation problems as shown in Table 5.5 and present instances of formulations of how conventional optimisation problems are linked to each other.

Facility Location Problem (FLP) and Job Assignment Problem (JAP)

We define FLP & JAP as follows: a set J of **n** jobs (this corresponds to l customers in FLP) have to be serviced on a chosen subset of K facilities, and this subset is assigned m agents

Table 5.5: Linked Problems

Optimisation Problems	FLP	KP	JAP	PFSP	TSP	QAP
FLP			FLP $\mapsto$ JAP	FLP $\mapsto$ PFSP		
KP			KP $\mapsto$ JAP	KP $\mapsto$ PFSP	KP $\mapsto$ TSP	
JAP					JAP $\mapsto$ TSP	
PFSP						
TSP						
QAP					QAP $\mapsto$ TSP	

to fulfil the jobs. Each facility can not service all the jobs. Once a job j is assigned to a facility $k, k = 1,2,\cdots,K$, it cannot be transferred to another facility as it must be completed at the assigned facility. Two decisions are incorporated when assigning the agents to the selected facilities; the decisions of what agent should be assigned to facility k, if selected, and what customer/job assigned to facility k should be serviced by an agent assigned to facility k. The processing time of job j by agent i is denoted as c_{ij}. The distance between each facility and a customer/job is denoted as d_{kj} and α_k denotes the running cost of facility k. We define D_j as the demand units of customer j, b_i as capacity of agent i and w_k as capacity of facility k. FLP & JAP consider a simultaneous minimisation of the cost of assigning agents to jobs and the cost of fulfilling the assigned jobs by the facilities. Here we define the linked problem of FLP & JAP as follows;

$$\begin{cases} minf(x_{FLP}) = \sum_{k=1}^{K} \alpha_k x_k + \sum_{k=1}^{K} \sum_{j=1}^{\mathbf{n}} d_{kj} x_k y_{kj} \\ minf^2(\mathrm{x}_{JAP\{x_{FLP}\}}) = \sum_{k=1}^{K} \sum_{i=1}^{m} \sum_{j=1}^{\mathbf{n}} c_{ij} x_k \mathbf{x}_{ki} y_{kj} z_{ij} \end{cases} \tag{5.23}$$

Subject to;

$$\sum_{k=1}^{K} \mathbf{x}_{ki} = 1 \quad \forall i = 1,2,\cdots m \tag{5.24}$$

$$\sum_{k=1}^{K} \sum_{i=1}^{m} \mathbf{x}_{ki} y_{kj} z_{ij} = 1 \quad \forall j = 1,2,\cdots \mathbf{n} \tag{5.25}$$

$$\sum_{j=1}^{\mathbf{n}} D_j \leq w_k \quad \forall k \tag{5.26}$$

$$\sum_{k=1}^{K} \sum_{j=1}^{\mathbf{n}} D_j \mathbf{x}_{ki} z_{ij} \leq b_i \quad \forall i \tag{5.27}$$

$$x_k, \mathbf{x}_{ki} y_{kj}, z_{ij} \in \{0,1\} \quad \forall i,j,k \tag{5.28}$$

Constraint 5.24 ensures that each job is assigned to precisely one facility, Constraints 5.25 each job is fulfilled by one facility and one agent. Constraint 5.26 ensures that the total demand of jobs assigned to each facility does not exceed the capacity of the facility. Constraints 5.27 ensures that the total resource requirement or total demand of jobs assigned to

each agent does not exceed the agent's capacity, and Constraints 5.28 defines the decision variables.

To further explain the linked problem of FLP & JAP, we define the problem instance from Tables 5.1 and 5.3 with 8 jobs, 5 facilities, and 5 agents. Here, C_n in Table 5.1 corresponds to J_n in Table 5.3. Given that $x_{FLP} = \{11010\}$ as seen in Figure 5.9, this means that facilities 1, 2 and 4 have been selected, and assuming that customers 1, 3, 4 and 6 are allocated to facility 2, customers 7 and 8 are allocated to facility 1 and customers 2 and 5 are allocated to facility 4. Then, when x_{FLP} is used as input to instantiate the JAP, this influences the feasibility of solutions obtained in JAP. Assuming, the solution for JAP is $x_{JAP} = \{EADDABCC\}$, i.e., agent E is assigned to job C_1, agent A to jobs C_2 and C_5, agent D to jobs C_3 and C_4, agent B to job C_6 and agent C to jobs C_7 and C_8. The feasibility of a solution for JAP depends on the facilities that fulfil the set of job/customer demands. In Figure 5.9, as agent C is assigned to perform customers jobs C_7 and C_8 which are fulfilled at facility F_1, if the agent is assigned a job which is to be fulfilled at facility F_4, for instance, the solution becomes infeasible. This simply means that the linkage between FLP and JAP is that the solution of FLP constrains the JAP. The implication is that a solution of FLP dictates which region in the solution space of the JAP is feasible.

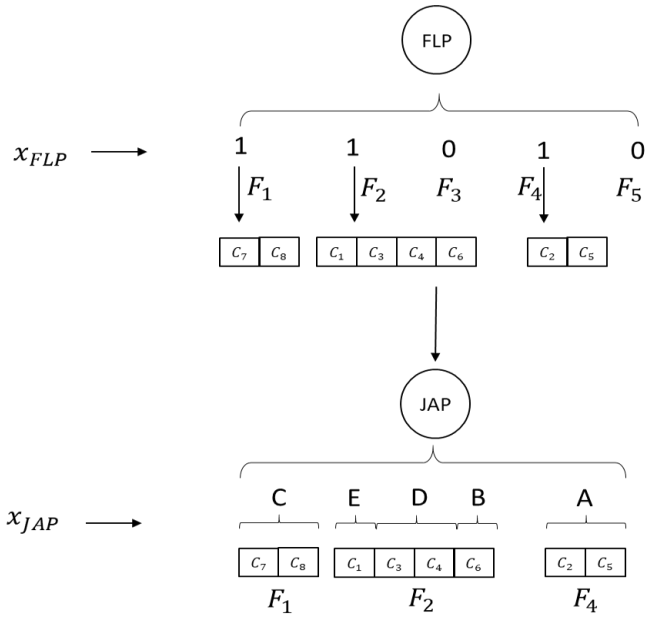


Figure 5.9: FLP & JAP Linked Problem Example

Facility Location Problem (FLP) and Permutation Flow Shop Scheduling Problem (PFSP)

FLP & PFSP can be defined as follows: a set J of l jobs (this corresponds to customers in FLP) have to be processed on a chosen subset of K facilities, and each facility contains the same set of m machines. Each facility is capable of processing all jobs. Once a job $j, j \in J$ is assigned to a facility $k, k = 1, 2, \cdots, K$, it cannot be transferred to another facil-

ity as it must be completed at the assigned facility. The processing time of job j on machine i at factory k is denoted as $\rho_{k,i,j}$. It is assumed that processing times do not change from facility to facility and therefore, $\rho_{1,i,j} = \rho_{2,i,j} = \cdots = \rho_{K,i,j}, \forall i = 1,2,\cdots,m$. We considered the minimisation of maximum makespan among selected factories and the minimisation of the cost of fulfilling the assigned jobs by the facilities. Given that the solution of FLP is $x_{FLP} = \{x_1,\cdots,x_K\} \in \{0,1\}^K$. This creates $\sum_{k=1}^{K} x_k$ PFSP problems, one for each of the facilities selected. Let $\sum_{k=1}^{K} x_k = s$ be the number of facilities opened/selected. Hence, $PFSP = \{PFSP_1, PFSP_2, \cdots, PFSP_s\}$.

PFSP is therefore minimised as $f(x_{PFSP}) = max(f(x_{PFSP_1}), f(x_{PFSP_2}), \cdots, f(x_{PFSP_s}))$. The linked model is as follows;

$$\begin{cases} minf^1(x_{FLP}) = \sum_{k=1}^{K} \alpha_k x_k + \sum_{k=1}^{K} \sum_{j=1}^{l} d_{kj} x_k y_{kj} \\ minf^2_{x_{FLP}}(x_{PFSP}) = \max\left\{C(x_{PFSP_k})x_k\right\}_{k=1}^{K} \end{cases} \tag{5.29}$$

For example, using the problem instances in Tables 5.1 and 5.4, given that $x_{FLP} = \{11010\}$, i.e., factories 1, 2 and 4 have been selected, and assuming that customers 1, 3, 4 and 6 are allocated to factory 2, customers 7 and 8 are allocated to factory 1 and customers 2 and 5 are allocated to factory 4. This then creates three sub-PFSP problems, as seen in Figure 5.10. The production plan is scheduled at the three selected facilities and then identifies the critical facility (i.e., the facility that produces maximum makespan). The implication of the linkages between FLP and PFSP results in changes in the solution structure and objective function of PFSP. A solution x_{FLP} of the FLP transforms the PFSP solution vector into s subvectors where s is the number of facilities selected in x_{FLP}.

Knapsack Problem (KP) and Job Assignment Problem (JAP)

The linked problem of KP & JAP involves selecting a subset of jobs that will maximise profit with a minimal cost associated with agents assigned to perform the selected jobs. Here, the number of items from the KP corresponds to the number of jobs in the JAP, i.e., n items = n. Each item has its value and weight. Given that the solution of the KP is $x_{KP} = \{x_1, x_2, \cdots, x_n\} \in \{0,1\}^n$. This changes the size of the JAP to $\sum_{i=1}^{n} x_i$. Assuming $\sum_{i=1}^{n} x_i = s$, then, JAP is minimised as $\sum_{i=1}^{n} \sum_{j=1}^{s} c_{ij} x_i$

$$\begin{cases} maxf(x_{KP}) = \sum_{i=1}^{n} \rho_i x_i \\ minf^2_{x_{KP}}(x_{JAP}) = \sum_{i=1}^{m} \sum_{j=1}^{\mathbf{n}} c_{ij} y_{ij} x_j \end{cases} \tag{5.30}$$

Lets assume that the solution of KP problem example in Table 5.2 is given as $x_{KP} = \{11101001\}$ which means that items 1, 2, 3, 5 and 8 are selected. When injected as input to the JAP problem example in Table 5.3, this solution creates a new instance of the JAP problem of size equal to the number of selected items in KP. This changes the dimensionality of the solution x_{JAP} of the JAP based on the number of items selected in the KP problem, as seen in Figure 5.11.

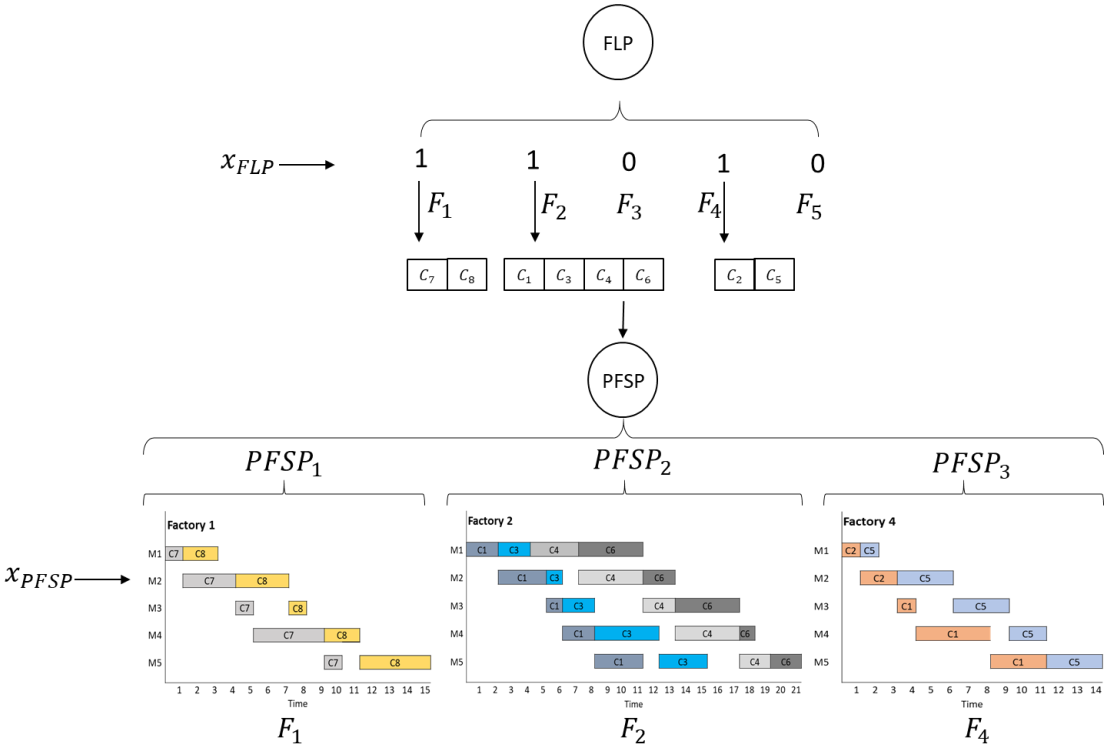


Figure 5.10: FLP & PFSP Linked Problem Example

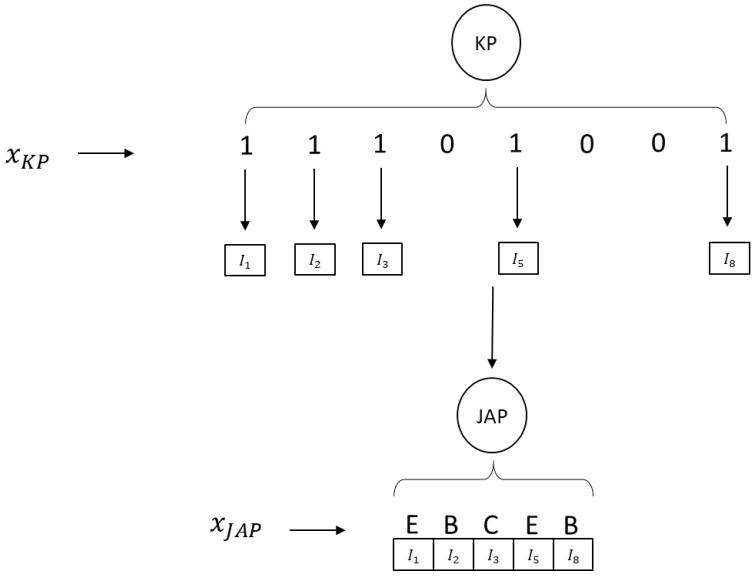


Figure 5.11: KP & JAP Linked Problem Example

Knapsack Problem (KP) and Permutation Flow Shop Scheduling Problem (PFSP)

KP & PFSP involves the selection of jobs for scheduling which will maximise profit at minimal completion time. We select from a subset of n items (KP) to be scheduled on m machines. Each item has its value and weight. Also, the number of items from the KP corresponds to the number of jobs in the PFSP. Given that the solution for KP is $x_{KP} = \{x_1, x_2, \cdots, x_n\} \in$

$\{0,1\}^n$, then, this changes the size of the PFSP to $\sum_{i=1}^{n} x_i$.

$$\begin{cases} maxf(x_{KP}) = \sum_{i=1}^{n} \rho_i x_i \\ minf^2_{x_{KP}}(x_{PFSP_s}) = C(x_{PFSP_s}) \end{cases} \tag{5.31}$$

$$\sum_{i=1}^{n} x_i = s \tag{5.32}$$

From Table 5.2, given that the solution for KP is $x_{KP} = \{11101001\}$, i.e., jobs 1, 2, 3, 5 and 8 are selected to be scheduled for production tasks for the PFSP in Table 5.4. The solution x_{KP} instantiates the PFSP in terms of the number of jobs selected by the KP. Figure 5.12 shows the example of the linkages between the KP and PFSP problems where the solution of the KP changes the size of the PFSP. The PFSP only schedules the selected items from the KP.

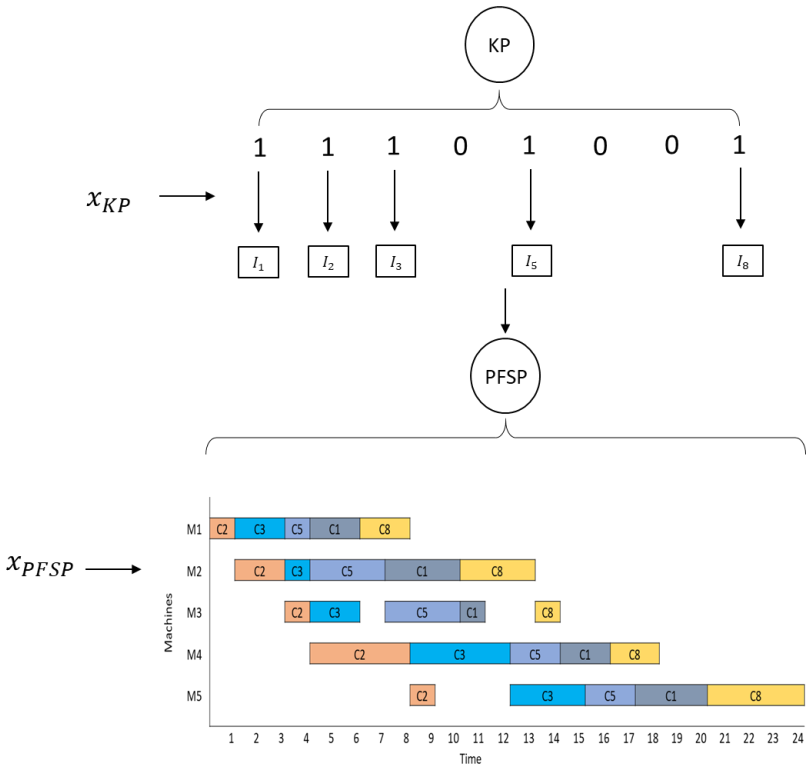


Figure 5.12: KP & PFSP Linked Problem Example

Knapsack Problem (KP) and Travelling Salesman Problem (TSP)

KP & TSP linked problem considers the selection of a subset of items to be picked by a travelling salesman. Each item has its value and weight, and the salesman's vehicle capacity corresponds to the knapsack's weight. In addition, the number of items from the KP corresponds to the number of locations of the TSP. Each item in the KP is associated with a location in the TSP. The solution of the KP is given as $x_{KP} = \{x_1, x_2, \cdots, x_n\} \in \{0,1\}^n$. This

changes the size of the TSP such that the subset of the item set is only visited. A typical solution pair representation is shown in Figure 5.13.

$$\begin{cases} maxf(x_{KP}) = \sum_{i=1}^{n} \rho_i x_i \\ minf^2_{x_{KP}}(x_{TSP_s}) = \sum_{j=2}^{s} d_{j-1,j} + d_{s,1} \end{cases} \tag{5.33}$$

Subject to;

$$\sum_{i=1}^{n} x_i = s \tag{5.34}$$

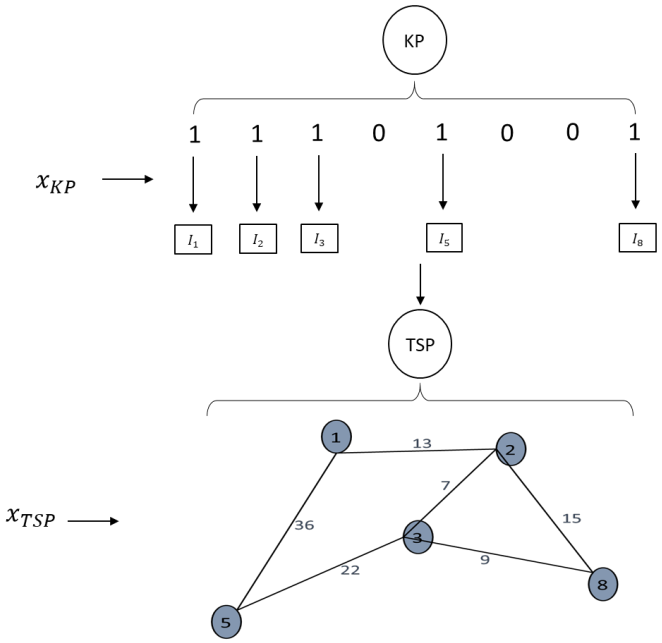


Figure 5.13: KP & TSP Linked Problem Example

Job Assignment Problem (JAP) and Travelling Salesman Problem (TSP)

The linked problem of JAP & TSP seeks to simultaneously minimise the cost of performing a set of jobs assigned to a set of agents and the total travelling distance of visiting the assigned job locations by the agents. In the linked problem, there are **n** cities with given distance matrix $\{d_{ij}\}$. Also, there are **n** jobs to be assigned to m agents/personnel with given availability/skill capacity/requirements. Here, the number of jobs for JAP corresponds to the number of locations in TSP. Each job in the JAP has a location in TSP. Given that the solution of the JAP is $x_{JAP} = (x_j)_{j=1}^{\mathbf{n}}$, such that, $x_j \in \{1,2,\cdots,m\}$. This creates multiple TSP problems, each TSP per agent. Assuming m agents are scheduled for **n** jobs, then, TSP = $\{TSP_1, TSP_2, \cdots, TSP_m\}$ as seen in Figure 5.14. TSP is therefore minimised as $f^2_{x_{JAP}}(\mathbf{x}_{TSP}) = \sum_{i=1}^{m} f^2(TSP_i)$.

$$\begin{cases} minf^1(x_{JAP}) = \sum_{i=1}^{m} \sum_{j=1}^{\mathbf{n}} c_{ij} y_{ij} \\ minf^2_{x_{JAP}}(\mathbf{x}_{TSP}) = \sum_{i=1}^{m} f^2(\mathbf{x}_{TSP_i}) \end{cases} \tag{5.35}$$

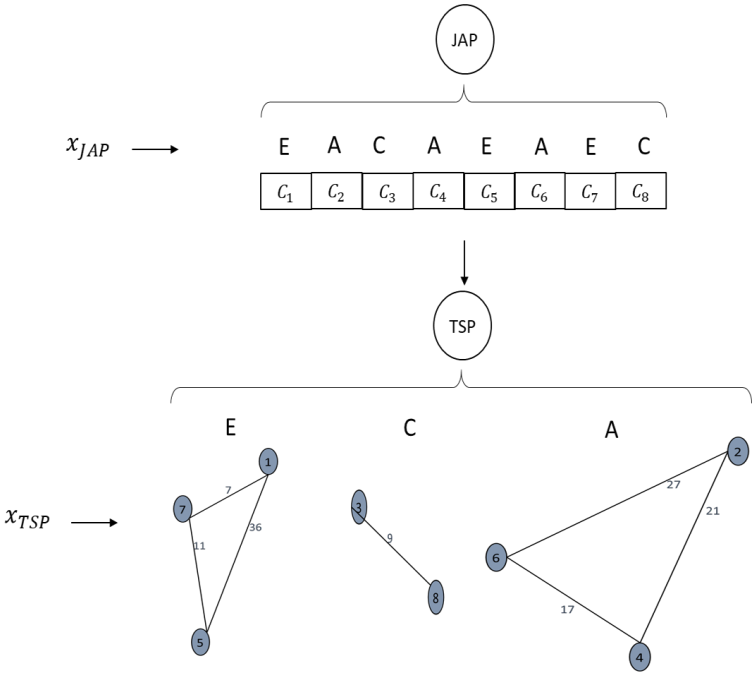


Figure 5.14: JAP & TSP Linked Problem Example

Quadratic Assignment Problem (QAP) and Travelling Salesman Problem (TSP)

The linked problem between QAP & TSP is the integration of assignment and sequencing problems prevalent in real-life situations in the supply chain involving a wide range of planning problems [197]. Let us consider a group of customers with unique products which a salesman delivers. Assuming each product is stored in different warehouses, i.e., a warehouse can only store a particular type of product. This means n warehouses are needed for n types of products. The linked problem is stated as follows: A salesman moves from the starting warehouse to deliver to the customer assigned to the warehouse, then go back to another warehouse to pick another product type for the next customer and travel to deliver the product at another location. The process continues until the last delivery is completed, and the salesman returns to the starting warehouse.

The objective of the linked problem is to minimise the salesman's total travelling distance, including the distances covered from each warehouse to the customer location and the distances from each customer location to each warehouse. We formulate the problem as follows;

$$minf(x_{QAP}, x_{TSP}) = \sum_{i=0}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\left(d_{ij} + d_{jk}\right)\alpha_{ijk} + d_{n0} \tag{5.36}$$

Subject to;

$$\sum_{i=0}^{n}\sum_{j=1}^{n}\alpha_{ijk} = 1 \quad \forall k = 1, \cdots, n \tag{5.37}$$

$$\sum_{i=0}^{n}\sum_{k=1}^{n}\alpha_{ijk}=1 \quad \forall j=1,\cdots,n \tag{5.38}$$

$$\sum_{j=1}^{n}\sum_{k=1}^{n}\alpha_{ijk}=1 \quad \forall i=0,1,\cdots,n \tag{5.39}$$

$$\alpha_{ijk}\in\{0,1\} \quad \forall i,j,k=1,\cdots,n \tag{5.40}$$

Constraints 5.37 - 5.39 ensure that each customer location and each warehouse is visited once. Constraint 5.40 is a decision variable declaration. The example shown in Figure 5.15 explains the sequencing of travel by the salesman while simultaneously maintaining the assignment of each customer to each warehouse. The permutation in the TSP adheres to the assignment problem, i.e., the assignment of each customer to the individual warehouse does not change while sequencing the salesman trip.

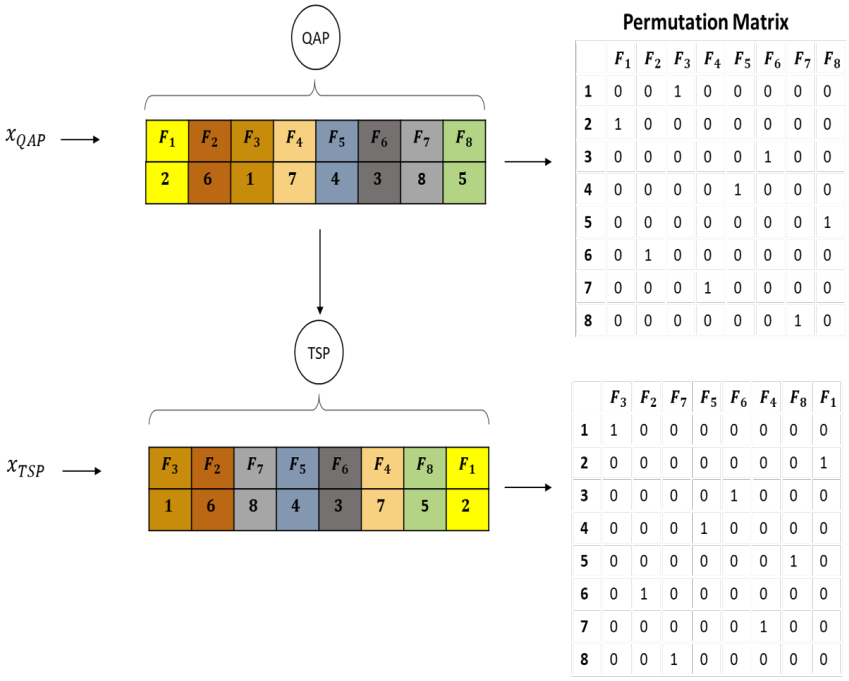


Figure 5.15: QAP & TSP Linked Problem Example

The implementation of the linked problems and formulations presented above are based on an object-oriented architectural framework which is presented in the next subsection.

5.4.3 Linked Problem Implementation Framework

The architectural framework of the linked problem is designed following an object-oriented architecture based on the jMetal multi-objective framework [119]. The framework contains a set of base classes which can be reused to design a new algorithmic methodology for linked problems and to implement complex operators for linked problems. We present the core classes of the linked problem implementation as shown in Figure 5.16. Figure 5.16 presents an UML diagram of core classes for the linked problem implementation. The principle of

the framework is that an algorithmic approach uses one or more algorithms to solve a linked problem. These are represented in classes, as seen in the figure.

From the concept of jMetal, We adopt a generic way to represent the core classes like Algorithm, AlgorithmLP, Problem, Operator, and Solution. These classes are superclasses which can be subdivided into functional or application-specific classes. For instance, the class Algorithm represents the superclass for individual solvers where different categories of metaheuristics can inherit it. Also, these classes are defined by the jMetal framework, but the functionalities were extended for the linked optimisation framework.

Class AlgorithmLP represents the superclass for all algorithmic approaches for linked optimisation problems. In Figure 5.16, three examples of algorithmic approaches have been implemented, including; Sequential, MCRGALP, and NSGALP. Different algorithmic approaches can inherit the AlgorithmLP superclass, and these approaches can take on one or more metaheuristics. An instance object of ALgorithmLP may require some application-specific parameters and can also embed some multi-objective properties or multi-criteria decision-making methods. The main method in AlgorithmLP is run(), which triggers the execution of the algorithmic approach selected.

In the implementation, all the problems have to inherit a generic superclass Problem. This class contains evaluate(), evaluateConstraints() and an additional method parentSolution(). The evaluate() method takes a candidate solution and evaluates the solution. The evaluateConstraints() also takes the candidate solution and checks for constraint violations. The parentSolution() method allows a problem, if dependent, when incorporated in a linked structure, to take as input a solution from a principal (parent) problem via a subclass of the AlgorithmLP. The parentSolution() is used to instantiate the problem based on the solution of the parent problem.

One of the core classes for this framework is the LinkedProblem superclass which takes two or more problems and contains a graph of the link structure based on an adjacency matrix. The two main methods used in this class are getLinkedProblem() and getAdjacencyMatrix(). The getLinkedProblem() retrieves the problem in the linked structure, and the getAdjacencyMatrix() retrieves the links between the problems in the linked structure. The implementation framework has implemented a set of linkages between two classical problem classes. Each linked problem uses an abstract of the LinkedProblem superclass to define the linkages.

The generic Solution superclass allows different types of solution objects. Three solution representation types are implemented - integer permutation, integer solution, and binary solution. Many more representations can be incorporated into the implementation framework by inheriting the Solution superclass.

The operator is a superclass that serves as the generic operator used by the different algorithms. The algorithm contains the getParameter() and setParameter() methods, which are used for adding and accessing operator-specific parameters. The generic object allows a high degree of flexibility regarding different operators' behaviours and adaptation.

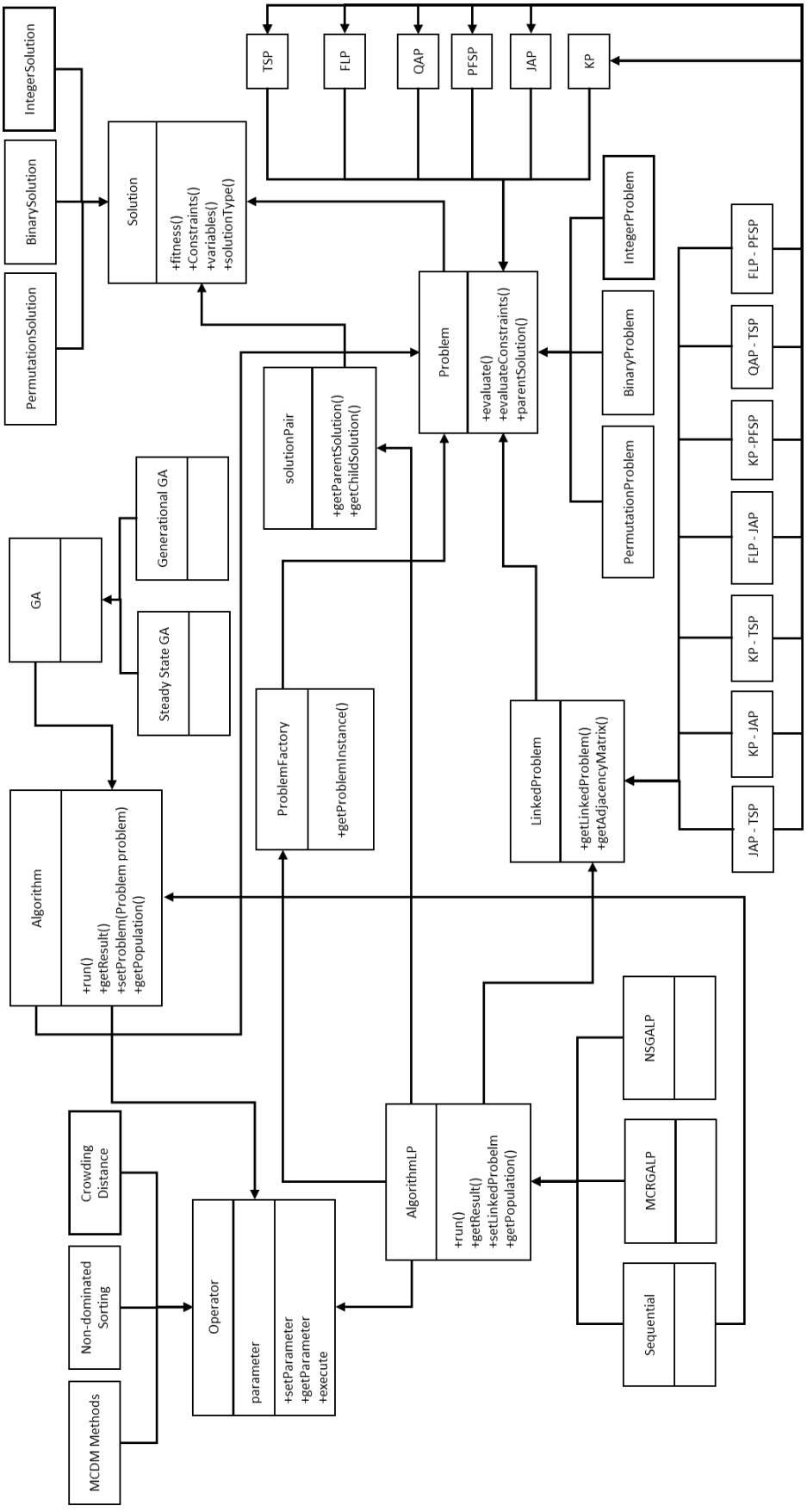


Figure 5.16: Linked Problem Implementation Architecture adapted from [119]

5.5 Experimentation and Analysis

An exploratory, experimental analysis is performed to ascertain the interactions between the selected problem combinations. In the data exploration analysis, we generate randomly and evaluate 10000 solutions for the parent problem. We instantiate the child problem for each solution generated on the parent problem based on our linked optimisation framework. We then generate randomly and evaluate 1000 solutions for each instantiated child problem and compute the mean value of the fitness values of the child solutions. Next, we determine the relationship between the sub-problems for each linked problem instance using Spearman's correlation coefficient. See Figure 5.17 showing the correlation analysis of individual linked problems. Computational analysis is performed on the same computer environment with Intel Core i9, 2.4GHz, 32GB RAM, and Windows 10 Enterprise OS. The experimental computation is implemented in Java.

5.5.1 Benchmarks

Table 5.6 presents benchmark instances of linked problem mix. The parent benchmark is the independent benchmark problem, and the child is the dependent benchmark problem. The table shows the total number of combined instances per problem mix and the range of problem sizes.

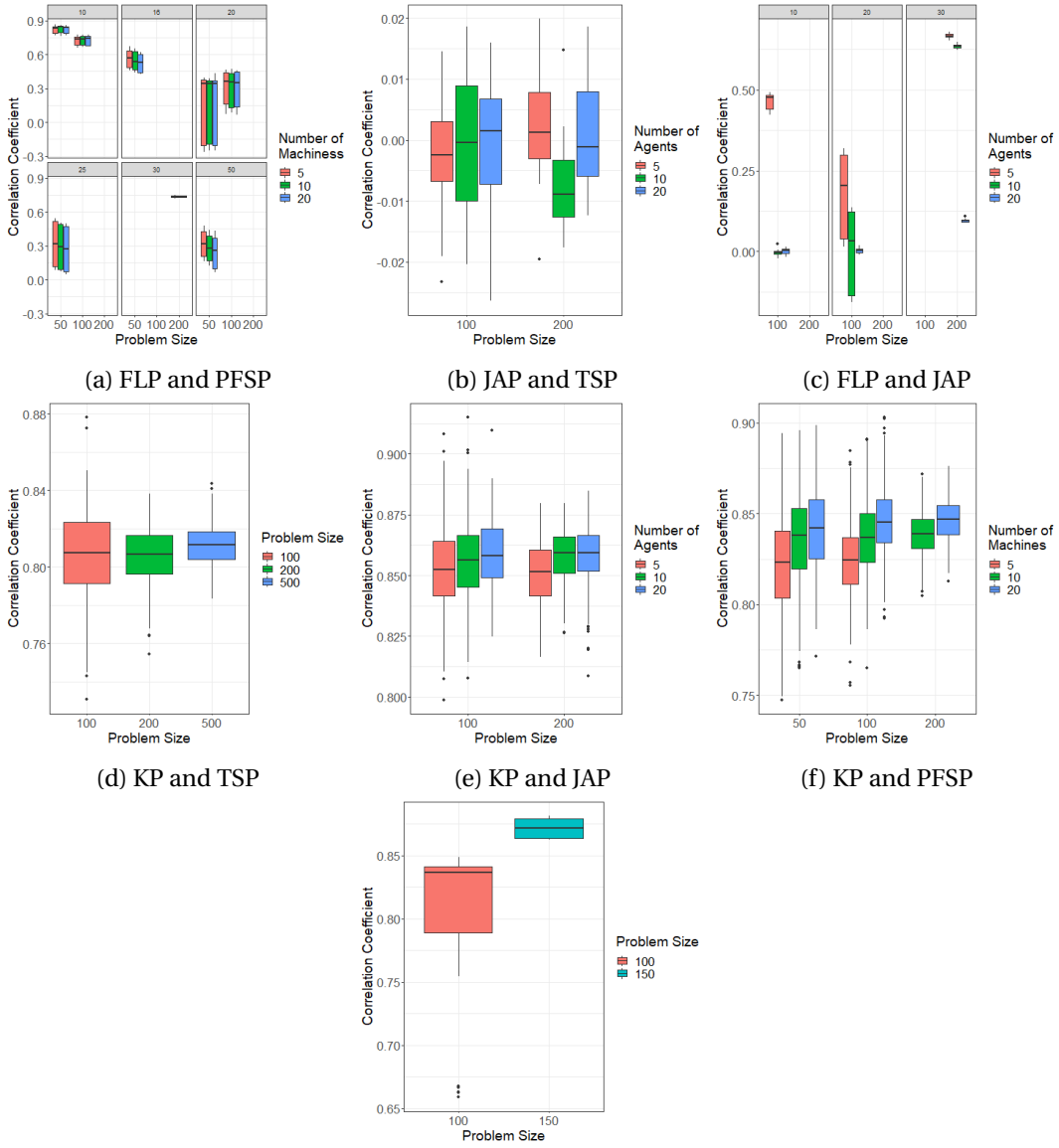
Table 5.6: Problem Mix Benchmark

Linked Problems	Parent Benchmark	Child Benchmark	No. of Instances	Range of Problem Size
FLP $\mapsto$ JAP	Beasley [24] and Holmberg [175]	Beasley [24]	144	100 - 200
KP $\mapsto$ JAP	KPLIB [208]	Beasley [24]	1700	100 - 200
FLP $\mapsto$ PFSP	Beasley [24] and Holmberg [175]	Taillard [343]	620	50 - 200
KP $\mapsto$ PFSP	KPLIB [208]	Taillard [343]	4008	50 - 200
KP $\mapsto$ TSP	KPLIB [208]	Gerhard [313]	737	50 - 500
JAP $\mapsto$ TSP	Beasley [24]	Gerhard [313]	114	100 - 200
QAP $\mapsto$ TSP	QAPLIB [51]	Gerhard [313]	60	100 - 150

5.5.2 Analysis of Results

We believe that selecting an appropriate approach for a linked optimisation problem is highly dependent on the level of relationship between the problems. We determine the relationship between the sub-problems in the data exploration analysis using Spearman's correlation coefficient. The correlation distribution of the linked problem under consideration is presented in 5.17. The degree of relationship between the mix of problems is a function of how a solution of one problem changes/modifies the solution, objective function and/or the constraint of another problem. A highly correlated problem mix may require

a robust approach that considers solving the problems simultaneously. A weak or uncorrelated relationship between the problem mix might require a simple and cheap approach that considers solving the problems in sequence.



(a) FLP and PFSP

(b) JAP and TSP

(c) FLP and JAP

(d) KP and TSP

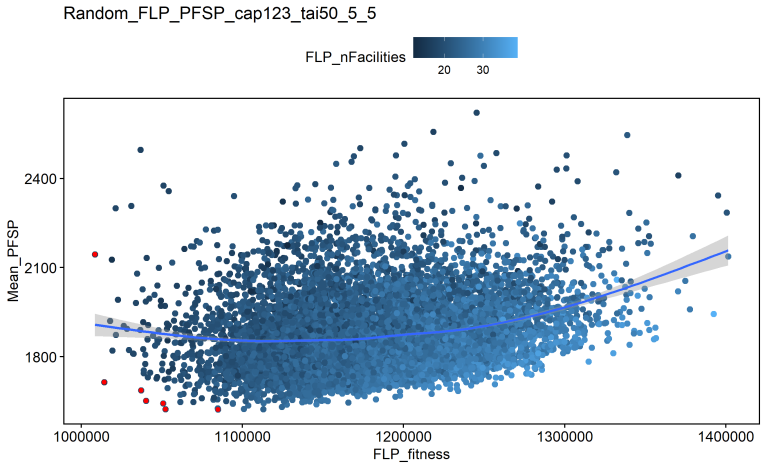
(e) KP and JAP

(f) KP and PFSP

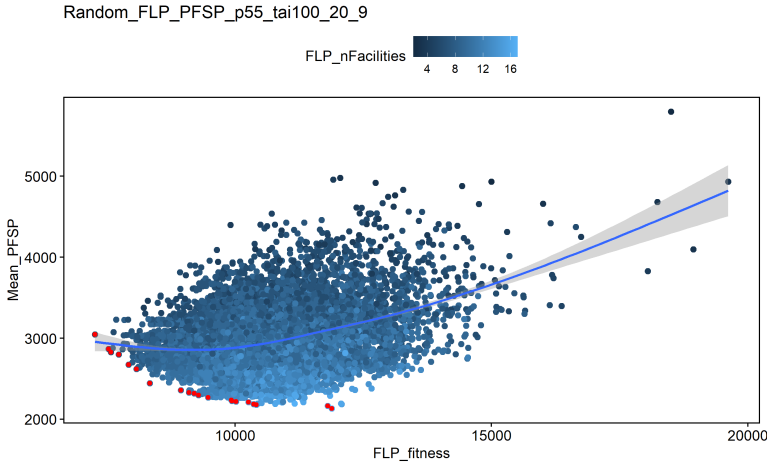
(g) QAP and TSP

Figure 5.17: Correlation Analysis of Linked Problems

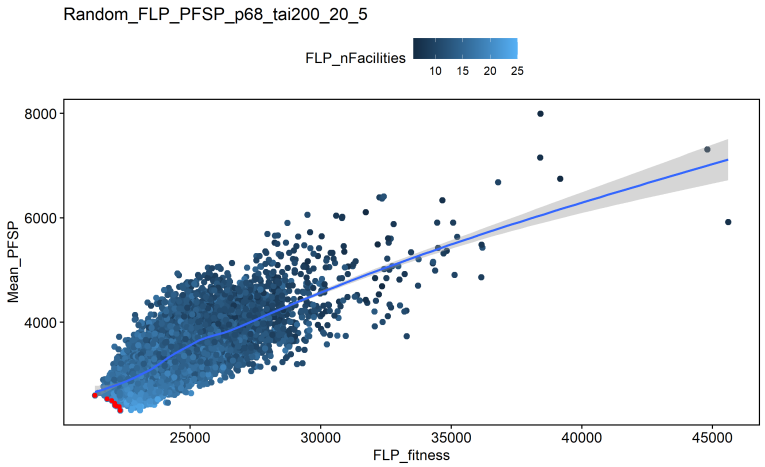
Figure 5.17a shows the correlation analysis of FLP & PFSP linked problems with a range of problem sizes between 50 - 200. The figure clearly shows that the highly correlated problem mixes comprise 10 facilities followed by 16 facilities. The correlation score decreases as the number of facilities increases. There is a slight or minimum impact of the number of machines on the correlation analysis. This suggests that, from the correlation distribution, the differences in correlation values are driven by the number of facilities for each problem instance data. This simply means that any algorithmic method used in tackling the problem



(a) 50 Facilities, 5 Machines & 50 Customers



(b) 16 Facilities, 20 Machines & 100 Customers

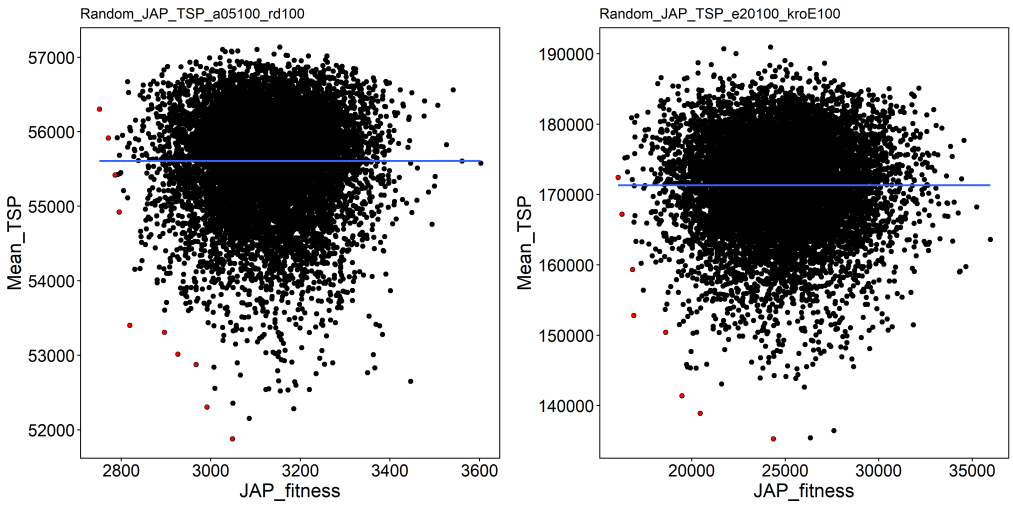


(c) 25 Facilities, 20 Machines & 200 Customers

Figure 5.18: FLP and PFSP Linked Problems

mix would perform differently from the basis of the different correlation values. The significance of this is that determining the underlying level of relationships in the linked problems

can influence the complexity of algorithmic design for such problems. Another way to look at the relationship between the FLP & PFSP is to identify the changes that occurred to the attributes of the dependent problem (PFSP) due to the linked structure formation. In this case, the changes happen at the objective function level. We only considered the facility (i.e., critical factory) with the highest makespan as the objective function to be minimised. This only considers the jobs assigned to the critical factory. However, it is quite different when the PFSP is considered in isolation as all the jobs' processing time is considered. Figure 5.18 presents three scatter plots showing the patterns of the association between the two problems. The three plots indicate a non-linear relationship between FLP & PFSP. This clearly suggests that selecting or designing an appropriate algorithmic approach to solve such linked problem must consider a robust and adaptive non-linear strategy.



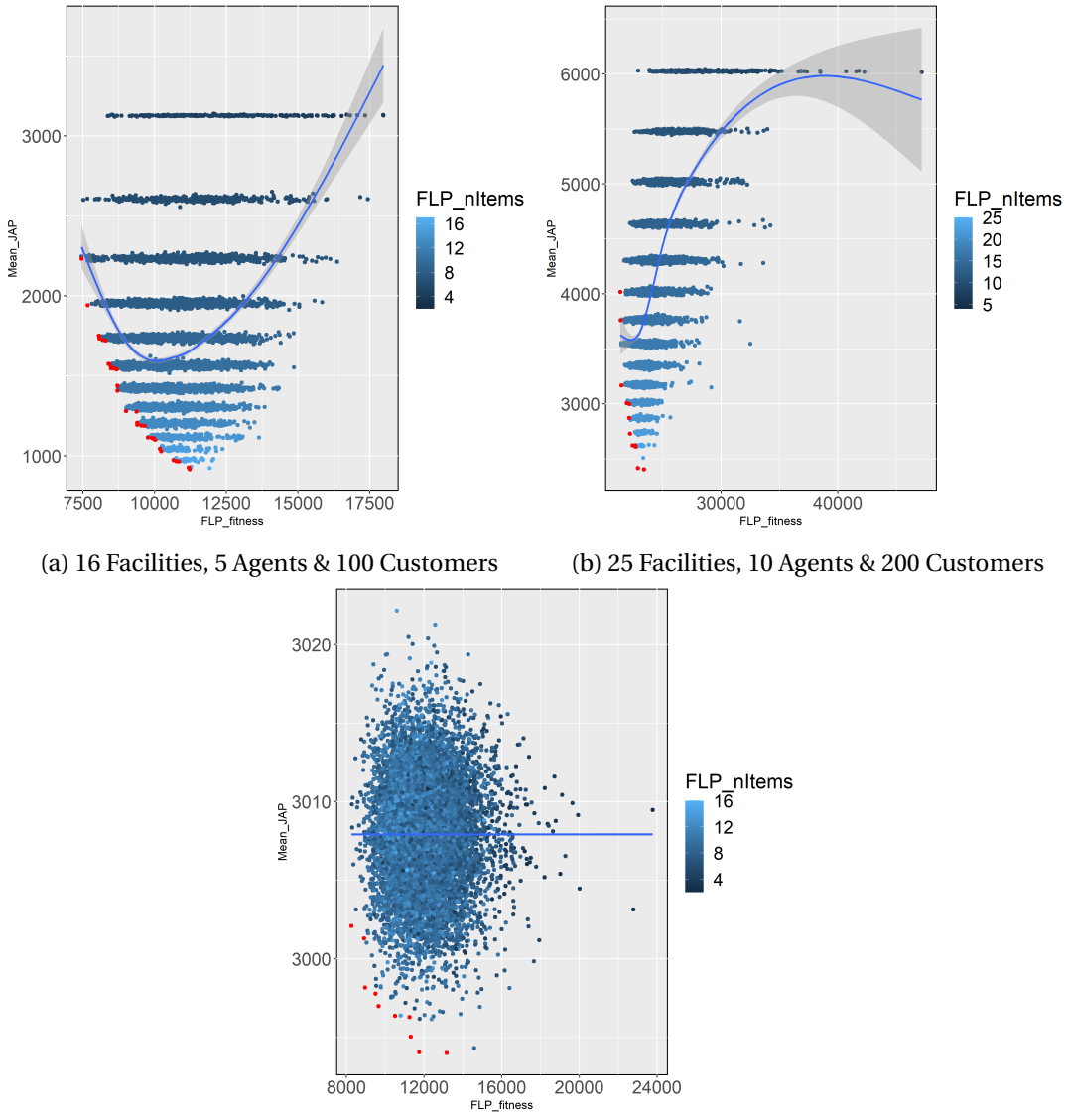
(a) 5 Agents & 100 Customers (b) 20 Agents & 100 Customers

Figure 5.19: JAP and TSP Linked Problems

The correlation distribution shown in Figure 5.17b for the JAP & TSP linked problem shows that there are no significant correlation between the corresponding objectives of the JAP and the TSP. In this case, simple algorithmic methods can be adopted to tackle the linked problem where both problems in the linked structure can be solved close to optimality. It is also important to note that linked problems of this nature are often caused by how a principal problem constrains its successor. Here, the capacity of the individual agents constrains the number of locations to visit in the linked problem. As suggested by the correlation distribution, the objective function obtained for the correlation analysis does not have a bearing on the constraints of the linked problem. In addition, Figure 5.19 shows clearly from the two plots the randomness between JAP & TSP. It simply means that no apparent relationship is evident from the data regarding the fitness values.

In Figure 5.17c, we consider the linkages between FLP & JAP with problem sizes between 100 and 200. Correlation distribution suggests that linked problem instances of 30

facilities are highly correlated compared with 10 and 20 facilities. The correlation distribution also suggests the influence of information associated with the individual problem instances. Different algorithmic approaches will perform differently regarding their ability to cope with the variability of the linked problem data. As seen in Figure 5.20, three scatter plots are shown. The three plots behaved differently, showing varying properties. An appropriate algorithmic strategy for such linked problem instances must be able to accommodate the different varying attributes in the problem data.



(a) 16 Facilities, 5 Agents & 100 Customers

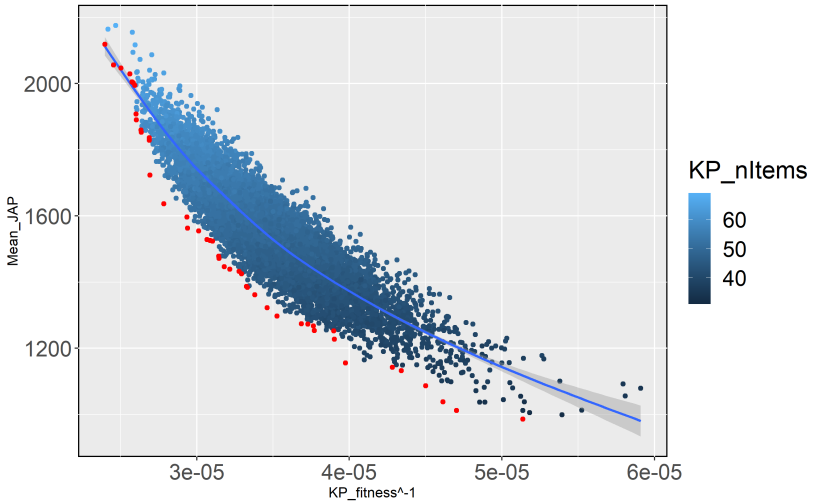
(b) 25 Facilities, 10 Agents & 200 Customers

(c) 16 Facilities, 20 Agents & 100 Customers

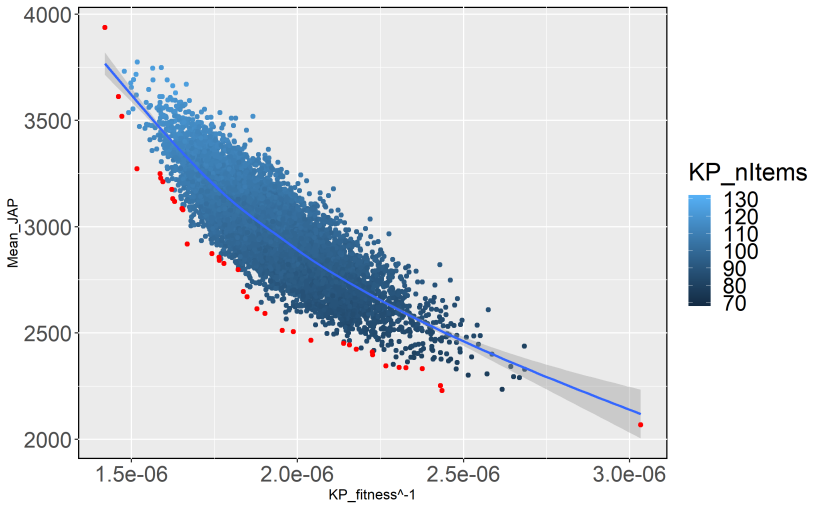
Figure 5.20: FLP and JAP Linked Problems

Figures 5.17d, 5.17e, 5.17f show consistent correlation distribution across the linked problem instances. The three linked problems (i.e., KP & JAP, KP & PFSP, and KP & TSP) show highly correlated values, which suggests strong and obvious evidence of linkages. The

KP problem serves as the principal problem, which instantiates the successor problems by changing the dimension of the solution space in the three examples of the linked problem.



(a) 5 Agents & 100 Items

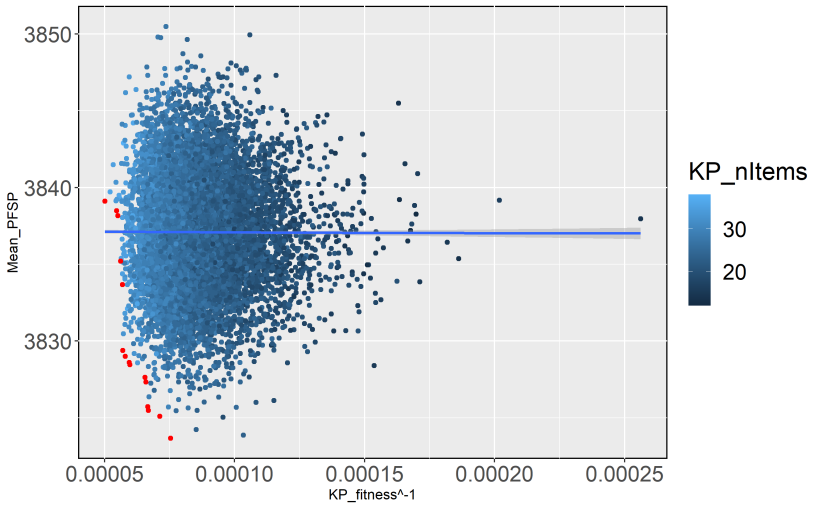


(b) 5 Agents & 200 Items

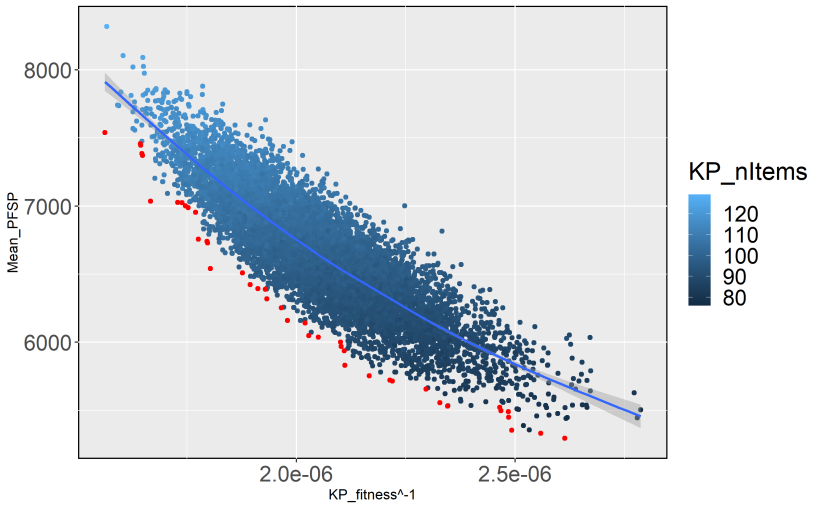
Figure 5.21: KP and JAP Linked Problems

We present three scatter plots of the linked problem in Figures 5.21, 5.22, and 5.23. We converted the KP maximisation problem to a minimisation problem by reciprocating the fitness value of KP. Figure 5.21 shows two scatter plots of fitness values of 1/KP minimisation problem vs mean fitness values of 1000 randomly generated solutions of instantiated JAP. The scatter plots show significant and indirect relationships between KP and JAP. This implies that changes in the fitness of KP have implications on the cost associated with performing the jobs by the agents. The relationship is linear and consistent across all instances of the KP & JAP linked problem. An algorithmic strategy to solve this linked problem must account for the linearity between the problems.

Figure 5.22 shows two scatter plots of fitness values of 1/KP minimisation problem vs



(a) 10 Machines & 50 Items



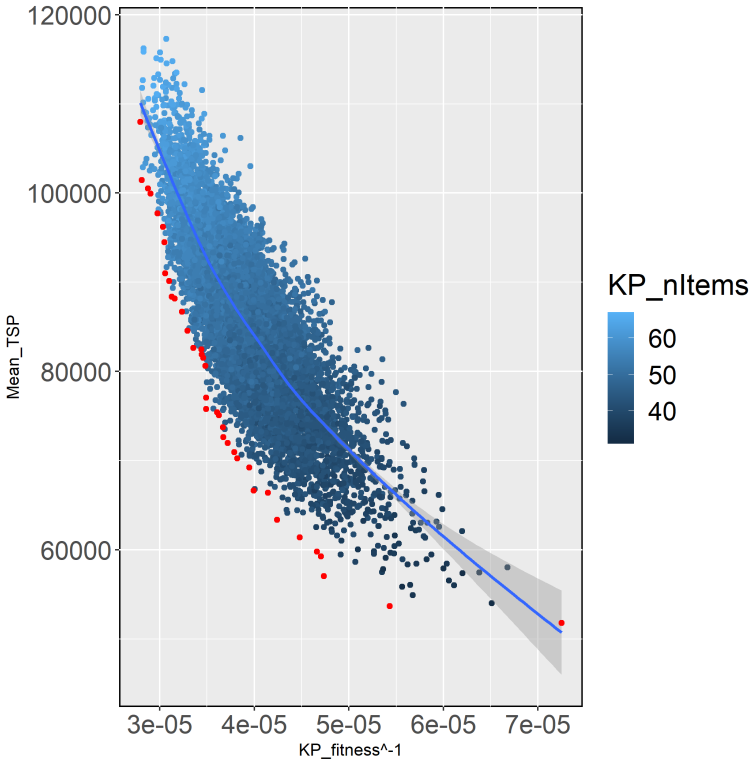
(b) 10 Machines & 200 Items

Figure 5.22: KP and PFSP Linked Problems

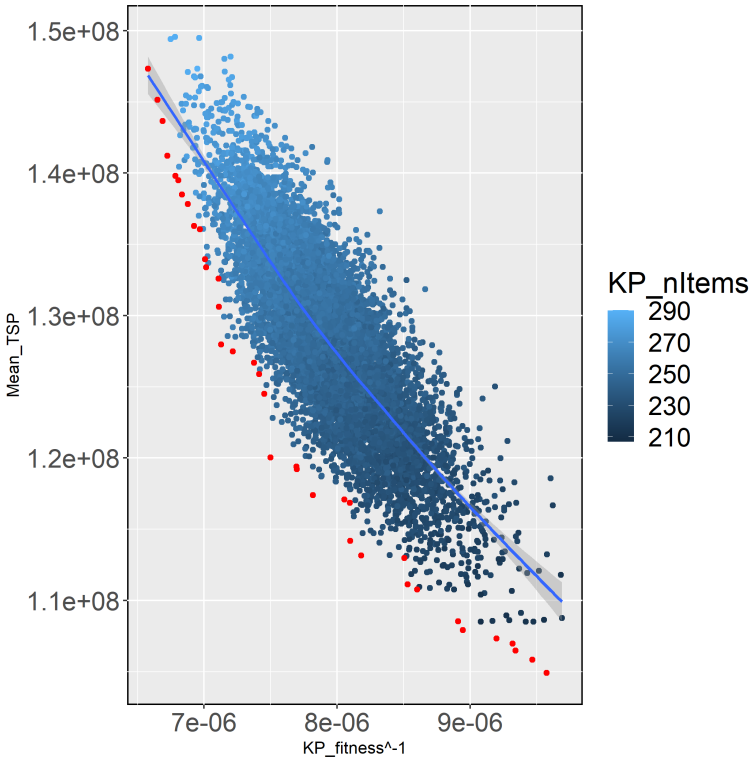
mean fitness values of 1000 randomly generated solutions of instantiated PFSP. Both scatter plots show differing patterns indicating that various attributes contribute to the variation in relationships. However, to a large degree, most instances show a significant and indirect relationship between the KP & PFSP linked problem. Therefore, maximising the fitness of KP results in increasing job completion time. The variability could be influenced by many factors, like the machines' attributes. Therefore, any algorithmic strategy must consider the variability in the linked problem instances.

The scatter plots in Figure 5.23 show a conflicting relationship, which is consistent across all linked problem instances. Hence, with more items selected in the KP problem, the salesman will have to travel to the locations where these items are, thereby increasing total travel time. A robust algorithmic strategy is necessary to account for the conflicting objectives.

In terms of the QAP & TSP linked problem, correlation distribution in Figure 5.17f re-



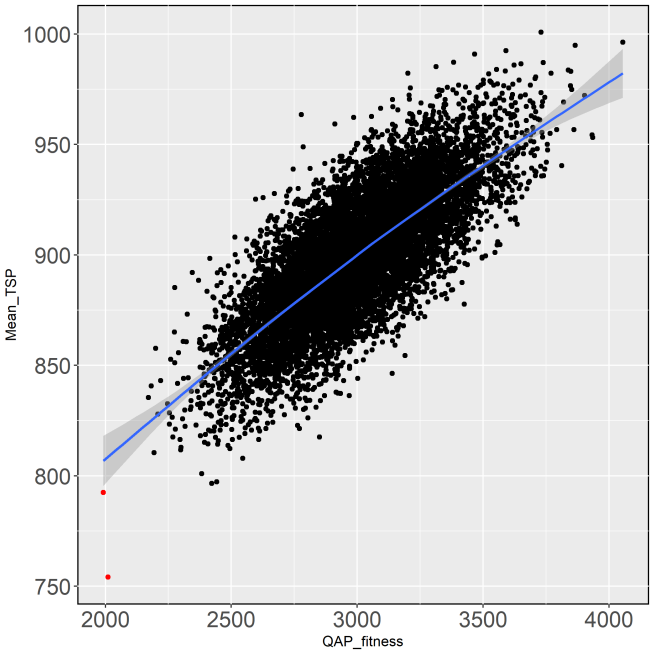
(a) 100 Items/Customers



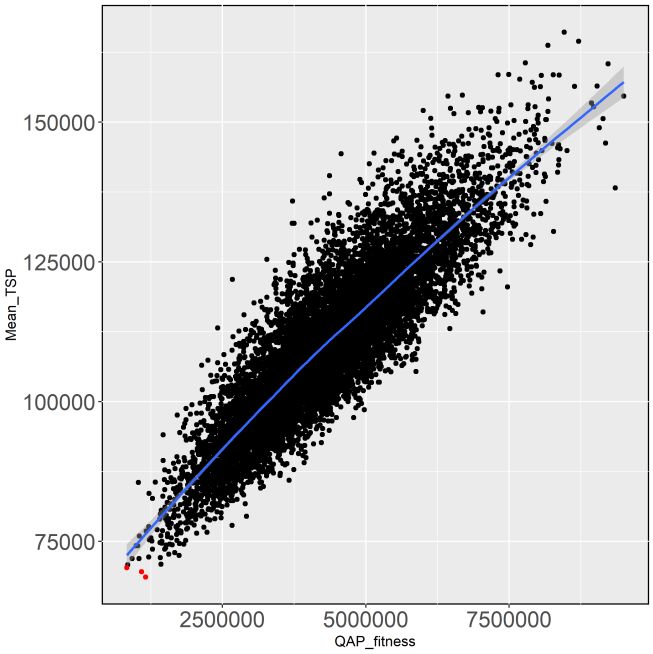
(b) 500 Items/Customers

Figure 5.23: KP and TSP Linked Problems

vealed a high level of correlation across all the linked problem instances. In addition, the two scatter plots shown in Figure 5.24 suggest a direct linear relationship across all the linked problem instances. On the occasion of a direct linear relationship in this type of linked problem, an algorithmic strategy could be adopted as long as the algorithm can achieve a close to optimal solution for the principal problem with negligible execution time.



(a) 100 Facility & Customers



(b) 150 Facility & Customers

Figure 5.24: QAP and TSP Linked Problems

The analysis of the simple experimental undertaking on the different mix of linked optimisation problems suggests that network interactions vary in a real-world supply chain. The interaction level occurs in three modes; the interaction that changes the solution representation of a problem, the interaction that changes the fitness function, and the interaction that changes/modifies the problem constraints. These interactions suggest whether a relationship is linear or non-linear. A linear relationship is identified in linked problems where, in most cases, the dependent sub-problem's solution changes. It is seen in the changes in the problem's dimensionality or the structure of the solution representation of the dependent problem. In a non-linear relationship, we may observe the changes in the fitness function or the changes in constraints of the dependent problem. As the analysis suggests, the underlying modes of interactions require different algorithmic approaches to reach an overall objective. Therefore, it is essential to understand these modes of interaction. Then, from the characteristics that define these interactions, appropriate algorithmic strategies can be implemented in a way that accounts for such characteristics. In addition to the interactions, it is also observed that the data drives most variations in the individual linked sub-problem instances. There is a possibility that wrong problem instances have been combined. This suggests a comprehensive investigation of the linkages in data of these problem instances.

5.6 Summary

This chapter's contribution addresses identifying modes of linkages in real-world supply chains. It addresses the research question (RQ2) and relates to objective (O2). The contribution provides some insights into understanding linked problems in network interactions and general knowledge about complex interactions. It is observed that the supply chain constitutes several systems defined by a network. In the network, there are different modes of interaction, which suggests different implications regarding strategic decision-making approaches. In order to get a good understanding of the interactions, the chapter proposes a methodological framework to explore different modes of linkages that constitute interactions in supply chain decision problems. It presents an implementation design to tackle the decision problems associated with the supply chain network. We analysed seven mixes of linked problems to present insights into understanding linked problems from the perspective of supply chain interactions. Results confirmed that different interactions exist with varying levels of relationships which are highly driven by data. The findings suggest a comprehensive investigation of the linkages in data of such problem instances.

6

Optimisation of Supply Chain Problems: Two Case Studies

This chapter presents two case studies of linked problem systems to support the research models proposed in Chapter 5. Each case considers two interconnected supply chain decision problems and presents three algorithmic strategies in tackling them.

6.1 Introduction

Several research communities are increasingly studying multiple optimisation problems whose solutions interact, thereby leading researchers to consider various approaches to a joint solution. A linked optimisation problem explains the concept of joint optimisation task involving n (i.e. $n \geq 2$) interdependent problems, i.e., a decision made for one problem causes a ripple effect on other dependent problems. We can define the link between two problems provided that the decision made for one problem p_1, i.e. the chosen solution x^1_*, affects the other problem p_2. Thus, the decision made in p_1 can affect elements of p_2, its search space x^2, its evaluation function f^2, and/or its constraints c^2. The formalism of the linked optimisation framework is presented in Section 3.2.1 of Chapter 3.

Linked problems are mostly solved separately and sequentially, often leading to suboptimal joint solutions [274]. Many examples occur in the literature where particular linked problems are modelled, and solutions are developed to solve the problem. In such studies, the solution of one problem dictates the outcome of others. A practical example is identified in [341], where a decision support optimisation system is used for a complex logistics operation involving five related problems; facility location, truck scheduling problem, packing problem, driver selection and vehicle routing. Another instance of a linked optimisation problem is in Chen et al. [73], where authors identified a significant interrelationship between crane handling and truck transportation and tackled the problems by integrating them using linked measures.

Real-world problems, like supply chains, are systems characterised by such combination and interdependency, where some features of the sub-components of the problem are

linked [42]. A decision made at one point in the supply chain could have significant consequences that ripple through linked production and transportation systems. This implies that an optimal solution for individual operational supply chain components might not guarantee an optimal solution for the overall supply chain problem [363]. The two case studies considered in this thesis are (1) the linkages between a facility location problem (FLP) and permutation flow shop scheduling problem (PFSP) of a distributed manufacturing system DMS, and (2) the integration between job assignment problem (JAP) and travelling salesman problem (TSP) of a service chain system where service personnel are required to perform tasks at different locations.

6.2 Proposed Algorithmic Approach

This section proposes three approaches to tackle the two case studies. They include the Sequential approach, Nondominated Sorting Genetic Algorithm for Linked Problem (NSGALP) and Multi-Criteria Ranking Genetic Algorithm for Linked Problem (MCRGALP). The three approaches are described in Sections 6.2.2-6.2.4. The motivation for selecting these approaches is as follows: first, the characteristics of linked optimisation problems vary with different data instances. Thus, different behavioural algorithmic approaches can be explored to address and understand such differing characteristics. Second, linked problems exhibit linkages that change features like problem solutions, objective functions, and constraints. With these forms of linkages, selecting the right approaches is significant to the specific linked nature of the problem.

6.2.1 Algorithmic Approaches Vs Existing Approaches

Recently, there has been new development in the area of evolutionary computation which involves evolutionary multi-tasking. Evolutionary multi-tasking involves multiple optimisation tasks at a time that takes advantage of implicit knowledge transfer across diverse problems [387]. Gupta et al. [158] introduced a new category of such optimisation problem and labeled this as multifactorial optimisation (MFO). MFO is an evolutionary multi-tasking optimisation paradigm that is characterised by multiple search spaces which correspond to individual tasks with corresponding function landscape. In the same manner, linked optimisation problem can also be categorised as a problem based on the premise of evolutionary multi-tasking paradigm. However, there are fundamental disparities between the principles of the two paradigms. One of the recently proposed solver for MFO is multifactorial evolutionary algorithm (MFEA), which harnesses the genetic complementarity between tasks. However, this algorithm does not take into consideration the relationship between the optimisation tasks before multi-tasking [387]. The algorithms modelled for MFO problems employ single population in which each task in the MFO problem represents a contribution factor that influences the population evolution [348]. This is achieved

by considering unified representation (unified search space) for the tasks. Each population is focused on one task and transfer knowledge with other individuals that belongs to other tasks based on assortative mating and cultural transmission [348]. However, linked optimisation problem considers problem interdependency that changes the individual problem solution space, objective function and/or constraints. This set of relationship is modelled within the algorithmic framework allowing the transfer of solutions for one problem to instantiate the other problem while conducting the evolutionary process. The ultimate goal of any solver adopted for MFO is to fully and concurrently optimise each task (problem) based on implicit parallelism of population-based search. Whereas, in linked optimisation problems, the purpose of the three algorithmic methods proposed in this chapter is to efficiently consider solutions corresponding to individual problems or tasks that resolve conflicting objectives among conflicting objectives corresponding to individual tasks. The prescribed scope for MFO is not driven by the concept of Pareto optimality. Rather, its scope is to find the global optimum of at least one constitutive objective function [158]. However, the proposed algorithmic approaches presented in this chapter considers all the constitutive objectives that can translate into finding the global optimum.

In the field of multi-objective optimisation (MOO), in some cases, standard evolutionary algorithms (i.e., NSGA-II, MOEAs, etc) for multi-objective optimisation are applicable for the purpose of linked optimisation problems. The fundamental difference is the fact that while the MOO algorithms attempt to efficiently resolve conflicts among competing objectives of the same task or problem, the sequential approach, NSGASLP, and MCRGALP proposed for linked optimisation problem aim to leverage the conflicting objectives corresponding to the individual problems or tasks. In addition, the capabilities embedded in the three proposed approached can tackle multiple MOO problems. For instance, the sequential approach for multiple MOO problems can be applied by incorporating multiple NSGA-IIs or MOEAs.

6.2.2 Sequential Approach

The sequential algorithmic approach applied to each case study follows a sequential process that incorporate two algorithms. This approach is commonly known for solving linked problems in a hierarchical structure, usually, between two decision-makers [232] [182]. Bilevel optimisation problem is a typical example where sequential based approach has been used [73] [240] [356].

We express a sequential approach as $\left(A_{p_1}, A_{p_2} | p_1, p_2\right)$, where $\left(A_{p_1}, A_{p_2}\right)$ represents two algorithms used in the sequential approach and $\left(p_1, p_2\right)$ represents the two problems for each case study.

Algorithm 7 shows a sequential approach for solving each case study. First, algorithm A_{p_1} solves problem p_1, selects the best solution $x^*_{p_1}$ then, uses best solutions $x^*_{p_1}$ to instantiate the second problem p_2 based on their linkage structure. Next, the instantiated p_2 is

solved using algorithm A_{p_2} and then select best solution $\mathbf{x}^*_{p_2}$. In sequential approach, solution $x^*_{p_1}$ instantiates problem p_2 once to return best solution pair.

Algorithm 7: *SEQUENTIAL*

$x^*_{p_1} \leftarrow A_{p_1}|p_1$;
$\mathbf{x}^*_{p_2} \leftarrow A_{p_2}|(p_2, x^*_{p_1})$;
Result: $(x^*_{p_1}, \mathbf{x}^*_{p_2})$

6.2.3 Non-dominated Sorting Genetic Algorithm for Linked Problem (NSGALP)

To solve each case study, using a multi-objective approach, we consider the solutions of p_1 and p_2 as a joint solution and adapt the fast non-dominated sorting procedure, a fast crowding distance estimation procedure, and a simple crowding comparison operator based on NSGA-II framework [103]. More details about the NSGA-II algorithm can be found in [103]. Algorithm 8 shows a multi-objective framework we adopted in tackling each case study. First, we perform random initialisation of population $pop^0_{(p_1,p_2)}$ containing N pairs of solutions $(x_{p_1}, \mathbf{x}_{p_2})$. After that, we evaluate the population with the respective objective function associated with the problem type. Then, we assign rank to the population by sorting them into fronts $\mathscr{F} = (\mathscr{F}_1, \mathscr{F}_2, \cdots)$, where $\mathscr{F}_\alpha \in \mathscr{F}$. After that, we determine the diversity $\text{dist}^{\alpha,\beta}_{(p_1,p_2)}$ for each joint solution using density estimation shown in Eq. 6.1 in front $\mathscr{F}_\alpha$. α denotes an individual front, and β represents a joint solution in front $\mathscr{F}_\alpha$. Then, we applied crowded binary tournament selection to select a mating pool $\mathscr{R}^t_{(p_1,p_2)}$ of size N. The crowded binary tournament operator is a modified version of the binary tournament selector that incorporates ranking and diversity. Next, we generate N joint offspring of population $Q^t_{(p_1,p_2)}$ using crossover and mutation. The procedure for offspring generation is provided in Section 6.3.3. Next, we evaluate the newly generated offspring and then combine the offspring $Q^t_{(p_1,p_2)}$ with the parent population $pop^t_{(p_1,p_2)}$. We assign ranks to each joint solution in the combined population and compute their crowding distance. At this stage, we adopt the crowded-comparison operator $\prec$ to guide our selection process based on rank and crowded distance attributes. The crowded comparison operator compares two joint solutions used in the tournament selection. At the initial stage with population $pop^0_{(p_1,p_2)}$, we apply the crowded binary tournament selection, but in subsequent stages, we use the crowded comparison operator. The crowded comparison operator compares all the joint solutions in the combined population $\mathbf{pop}^t_{(p_1,p_2)}$ of size $2N$ and select new population $pop^{t+1}_{(p_1,p_2)}$ of size N. The procedure continues until the stopping criterion is met and then returns $\mathscr{F}_1$ as the Pareto optimal set.

Non-Dominated Sorting

We adopt [103] fast sort algorithm in NSGALP to sort initial population. In the sorting process, we create a set S that contains all the dominated individual solution pairs. The solution pairs left out form the first front. We then determine the individuals that dominate others in set S and place them in the next front. We continued the process until we find the subset of S where any other individual dominates no one individual.

Crowding Distance

Crowding distance defines the sum of individual distance between each Pareto solution on a front corresponding to each objective[103]. Crowding distance is assigned front-wise and allows comparison within individual fronts [277]. The crowding distance is calculated after a non-dominated sorting is completed. Each front is considered individually and then sorted in non-decreasing order of fitness value so that the first solution pair and the last solution pair are assigned infinite values. The crowding distance of each solution pair is calculated using Eq. 6.1.

$$\text{dist}^{\alpha,\beta}_{(p_1,p_2)} = \sum_{\iota=1}^{n} \frac{f^{\iota}_{\alpha,\beta+1} - f^{\iota}_{\alpha,\beta-1}}{f^{\iota}_{\alpha,max} - f^{\iota}_{\alpha,min}} \quad \forall \alpha \tag{6.1}$$

Algorithm 8: *NSGALP*

$pop^{0}_{(p_1,p_2)} \leftarrow$ Randomly initialise population ;
Fitness evaluation on $pop^{0}_{(p_1,p_2)}$;
Assign fast non-dominated sort to $pop^{0}_{(p_1,p_2)}$;
Apply crowding-distance assignment to $pop^{0}_{(p_1,p_2)}$;
$t \leftarrow 0$;
while *Stopping criterion not met* **do**
  $\mathscr{R}^{t}_{(p_1,p_2)} \leftarrow$ Select from $pop^{t}_{(p_1,p_2)}$;
  $Q^{t}_{(p_1,p_2)} \leftarrow$ Generate offspring from $\mathscr{R}^{t}_{(p_1,p_2)}$;
  Fitness evaluation on $Q^{t}_{(p_1,p_2)}$;
  $\mathbf{pop}^{t}_{(p_1,p_2)} \leftarrow pop^{t}_{(p_1,p_2)} \cup Q^{t}_{(p_1,p_2)}$;
  Assign fast non-dominated sort to $\mathbf{pop}^{t}_{(p_1,p_2)}$;
  Apply crowding-distance assignment to $\mathbf{pop}^{t}_{(p_1,p_2)}$;
  $pop^{t+1}_{(p_1,p_2)} \leftarrow$ Select survivor from $\mathbf{pop}^{t}_{(p_1,p_2)}$;
  $t \leftarrow t+1$;
end
Result: $\mathscr{F}^{1}_{(p_1,p_2)}$

6.2.4 Multi-Criteria Ranking Genetic Algorithm for Linked Problem (MCRGALP)

The multi-criteria approach applied to each case study uses a similar approach to the one outlined in Section 6.2.3 but with differences in the output returned and the comparison operators used in the tournament selection. Unlike NSGALP, MCRGALP uses a multi-criteria performance metric known as Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS) [378], which assigns a performance score to each joint solution. In the MCRGALP approach, we use TOPSIS as a comparison operator in the tournament selection process to guide the selection of joint solutions at different phases of the algorithm process. Details about the performance score of TOPSIS is provided in the subsection below. In this approach, each joint solution in population $pop^{t}_{(p_1,p_2)}$ is assigned a performance score. The joint solutions are sorted in descending order, i.e., the highest score indicates the best solution pair. The performance score achieved by a joint solution is bounded between 0 and 1. In the offspring generation phase, we adopt the procedure specified in Section 6.3.3 to obtain offspring population $Q^{t}_{(p_1,p_2)}$. Thereafter, we evaluate $Q^{t}_{(p_1,p_2)}$ and then, combine the offspring $Q^{t}_{(p_1,p_2)}$ with the old parent population $pop^{t}_{(p_1,p_2)}$. Next, we select a new population $pop^{t+1}_{(p_1,p_2)}$ using the TOPSIS score to sort solution pairs in descending order of performance score. This process repeats until a stopping criterion is met, and then, the solution pair $(x^{*}_{p_1}, \mathbf{x}^{*}_{p_2})$ with the highest score is selected.

Algorithm 9: *MCRGALP*

$pop^{0}_{(p_1,p_2)} \leftarrow$ Randomly initialise population ;
Fitness evaluation on $pop^{0}_{(p_1,p_2)}$;
$t \leftarrow 0$;
while *Stopping criterion not met* **do**
  Assign score to each solution pair in $pop^{t}_{(p_1,p_2)}$;
  $pop^{*}_{(p_1,p_2)} \leftarrow$ Get best pairs of $pop^{t}_{(p_1,p_2)}$;
  $Q^{t}_{(p_1,p_2)} \leftarrow$ Generate offspring from $pop^{*}_{(p_1,p_2)}$;
  Fitness evaluation on $Q_{(p_1,p_2)}$;
  Assign score to each solution pair in $Q_{(p_1,p_2)}$;
  $pop^{t}_{(p_1,p_2)} \leftarrow pop^{t}_{(p_1,p_2)} \cup Q^{t}_{(p_1,p_2)}$;
  $pop^{t+1}_{(p_1,p_2)} \leftarrow$ Get top N solution pairs with best score from $pop^{t}_{(p_1,p_2)}$;
  $t \leftarrow t+1$;
end
Result: $(x^{*}_{p_1}, \mathbf{x}^{*}_{p_2})$

Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS)

We adopted the TOPSIS method as a selection operator in MCRGALP. TOPSIS is one of the multiple criteria decision-making methods introduced by Yoon and Hwang [378]. TOPSIS

uses Euclidean distance computation to select alternatives. It considers alternatives that are closest to the positive ideal solution but farthest from the negative ideal solution from a geometrical point [304]. TOPSIS affords the decision-maker the nearest alternative, which is considered the best one according to the score, illustrated by the judgement [258]. TOPSIS judges from the perspective of each decision-maker. TOPSIS decision-making technique is classified into five main steps [353].

The first step involves normalising the decision matrix using Eq. 6.2. Let $|pop^t|$ denote the size of solution pairs in the current population pop^t and $f_{i\iota}$ represents each decision criterion such that $\iota = 1, \cdots, n$ (i.e., f_{i1} - facility cost and f_{i2} - makespan for $n = 2$).

$$r_{i\iota} = \frac{f_i^\iota}{\sum_{i=1}^{|pop^t|}(f_i^\iota)^2} \tag{6.2}$$

In the second step, we determine normalized weighted value $v_{i\iota}$ with given weight $w_\iota = (w_1, \cdots, w_n)$ using Eq. 6.3. w_j is the weight of the decision criterion for all ι and $\sum_{\iota=1} w_\iota = 1$.

$$v_{i\iota} = r_{i\iota} * w_\iota \tag{6.3}$$

Step three identifies the ideal best solutions and ideal worst solutions using Eq. 6.4 and Eq. 6.5. Set $I = \{\iota = 1, \cdots, n\}$ is associated with benefit criteria and set $I' = \{\iota = 1, \cdots, n\}$ is associated with cost.

$$v_\iota^+ = \left\{(\max v_{i\iota} | \iota \in I), (\min v_{i\iota} | \iota \in I'), i = 1, \cdots, |pop^t|\right\} = \left\{v_1^+, \cdots, v_n^+\right\} \tag{6.4}$$

$$v_\iota^- = \left\{(\min v_{i\iota} | \iota \in I), (\max v_{i\iota} | \iota \in I'), i = 1, \cdots, |pop^t|\right\} = \left\{v_1^-, \cdots, v_n^-\right\} \tag{6.5}$$

Step four measures the difference between each alternative to the ideal best solutions v_ι^+ and ideal worst solutions v_ι^- using Euclidean distance computation. Let S_i^+ be defined as an alternative distance from the ideal best solution and S_i^- as an alternative distance from the ideal worst solution using Eq. 6.6 and Eq. 6.7 respectively.

$$S_i^+ = \sqrt{\sum_{\iota=1}^{n}(v_{i\iota} - v_\iota^+)^2} \tag{6.6}$$

$$S_i^- = \sqrt{\sum_{\iota=1}^{n}(v_{i\iota} - v_\iota^-)^2} \tag{6.7}$$

where $i = 1, 2, \cdots, |pop^t|$

Step five calculates the performance score and ranks the solution pairs from the largest value to the smallest value. The alternative with the most significant value is the best solution pair. The performance score $\mathscr{P}_i$ for each solution pair is defined in Eq. 6.8.

$$\mathscr{P}_i = \frac{S_i^-}{S_i^+ + S_i^-} where \quad 0 \leq \mathscr{P}_i \leq 1 \tag{6.8}$$

6.3 Case Study 1: FLP (p_1) and PFSP (p_2)

Globalisation has created a dynamic market structure and rendered traditional centralised production scheduling inefficient in responding to changing market needs [246]. Distributed manufacturing systems have become more prevalent in today's supply chain because they can achieve lower production costs and management of risks [67, 107].

Distributed manufacturing involves assigning all production tasks among multiple factories and scheduling at the factories to optimise multiple objectives [246]. A prominent example is distributed permutation flow shop scheduling (DPFSP). DPFSP is applied in several fields such as petrochemical processing, automobile manufacturing, and cell manufacturing. It is an extension of the classical permutation flow shop scheduling problem PFSP [246]. In DPFSP, jobs are allocated among multiple factories, making DPFSP more complicated. However, DPFSP considers only the scheduling operation at the factories but ignores the cost associated with fulfilling the jobs by individual factories.

This first case study introduces a new variation of DPFSP, which accounts for the factory cost. We call this problem Facility Location Problem and Permutation Flow shop Scheduling Problem (FLPPFSP). This section is summarised as follows.

- We formulate a novel model of FLPPFSP to include the factory cost in fulfilling the jobs. To the best of our knowledge, no similar model has been published in previous research. This section investigates an FLPPFSP, rarely reported in the existing research. Most studies only consider the makespan achieved by the schedules but ignore the cost associated with fulfilling the jobs (distribution cost) by individual factories. We adopt a linked optimisation framework to represent the linkages between the two sub-problems.

- We propose a sequential approach, a multi-objective approach (NSGALP), and a multi-criteria decision-making (MCRGALP) algorithm. Interacting optimisation problems are connected in diverse ways and may require different and complex algorithmic designs to tackle them.

- We pre-assess the relationship between the two problems in FLPPFSP to guide our understanding of selecting an appropriate algorithmic approach.

6.3.1 Problem Background

This section briefly reviews related work of distributed permutation flow shop problem (DPFSP) and then introduces a new variation of DPFSP.

Distributed permutation flow shop scheduling problem (DPFSP) was first explored by [279], leading to six different mixed integer linear programming (MILP) models. The authors propose two simple factory assignment rules together with 14 heuristics. Gao and Chen

[144] proposed a local search-based genetic algorithm to solve DPFSP where the objective is to minimise the makespan.

A recent study of DPFSP by [246] involves using a knowledge-based multi-objective memetic optimisation algorithm (KMMOA) to address a sustainable distributed permutation flow-shop scheduling problem. The problem involves non-identical factory (DPFSP-NF) that minimises makespan, negative social impact (NSI), and total energy consumption (TEC). Another variation of DPFSP was proposed by [399] which considers a distributed assembly permutation flow shop scheduling problem (DAPFSP) by minimising makespan using a genetic algorithm. In [134], DPFSP was addressed to minimise the total flowtime using 18 constructive heuristics and an iterative improvement algorithm. Fu et al. [141] considered a stochastic multi-objective DPFSP involving total tardiness constraint in minimising makespan and total energy consumption.

DPFSP typically focuses on production benefits (i.e., minimising makespan, tardiness and total flowtime) relating to the flow shop scheduling and ignores the cost of fulfilling the customer demands/jobs by the factories and the proximity of factories to job location/delivery sites. In DPFSP, two decisions are taken: job assignment to factories and job scheduling at each factory [279]. However, for the high performance of DPFSP, an additional critical decision arises: selecting a set of factories that delivers cost benefits, reduces time to meet demand and reduces environmental impacts. A subset of factories might deliver cost savings and, at the same time, reduce job completion time. This, therefore, introduces an additional classical optimisation problem known as the facility location problem (FLP).

In FLP, the best factories are selected that satisfy constraints requiring jobs are serviced by the selected factories. The objective of the problem involves the selection of facilities to service a set of customers' demands that will minimise the total cost of fulfilling customer demands. Solving FLP requires two decisions: selection of facilities and demand/job assignment to facilities. FLP may be classified into capacitated and uncapacitated variants.

Adapting the FLP to the DPFSP creates a linked optimisation problem, where two classical optimisation problems (FLP and PFSP) are linked. This chapter, therefore, presents a novel model of a facility location problem and permutation flow shop scheduling problem (FLPPFSP) with identical factories. This means that job processing times on machines are the same from the factory to factory. Also, FLPPFSP considers a subset of facilities to process a sequence of jobs which seeks to minimise the facility cost and makespan simultaneously.

Similarly, [75] investigated three problems that involve the integration of order batching, batch sequencing and picker routing (IOBSPR) in warehouses. They proposed an algorithm linking hybrid-coded genetic algorithm and ant colony optimisation to tackle the IOBSPR model. It was argued that linking these problems can result in efficiency improvements and cost savings which, on average, can result in between 5% and 20% cost savings [274].

To further explain the FLPPFSP linked problem, we use problem instance of 8 jobs, 5 factories, and 5 machines as shown in Tables 6.1 and 6.2. The fixed cost of factories and total job cost per factory (as C_n is referred to a job in Table 6.2) are recorded in Table 6.1. So, for

Table 6.1: Example of FLP Instance

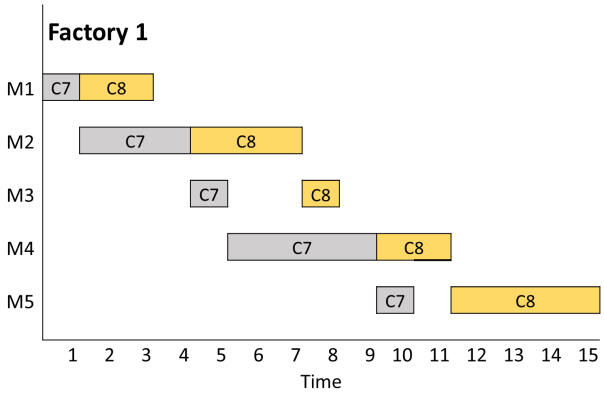
Fixed Cost	20	23	25	29	30
Demand cost to facility	F1	F2	F3	F4	F5
C1	34	20	76	54	44
C2	34	68	23	27	43
C3	56	20	65	53	29
C4	68	25	47	42	60
C5	45	33	65	10	36
C6	51	30	71	60	87
C7	59	35	81	50	65
C8	30	90	21	25	42

Table 6.2: Example of PFSP Instance

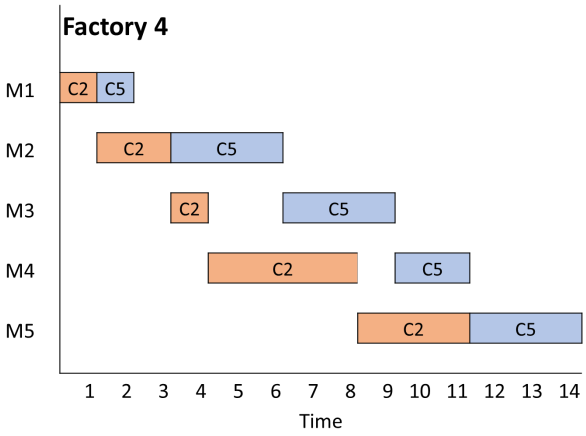
	Processing Time				
Jobs	Machine 1	Machine 2	Machine 3	Machine 4	Machine 5
C1	2	3	1	2	3
C2	1	2	1	4	1
C3	2	1	2	4	3
C4	3	4	2	4	2
C5	1	3	3	2	3
C6	4	2	4	1	2
C7	1	3	1	4	1
C8	2	3	1	2	4

instance, fulfilling job C_1 at facility 1 will cost 20 (the fixed cost) + 34 (the total demand cost). The processing time of jobs is recorded in Table 6.2. A solution for the FLP is $x_{FLP} = 11010$; this means that factories 1, 2 and 4 are chosen, and assuming that customers 7 and 8 are allocated to factory 1, customers 1, 3, 4 and 6 allocated to factory 2, and customers 2 and 5 allocated to factory 4. The calculation of the two objectives of the above solutions are as follows: the first objective - minimising facility cost of fulfilling jobs: $f(x_{FLP}) = (20+23+29)+(59+30+20+20+25+30+27+10) = 293$; the second objective - minimising makespan: $f(x_{PFSP}) = C_{max} = max\{C_1, C_2, C_4\} = 21$. Figure 6.1 shows the Gantt chart of the solution for the PFSP problem.

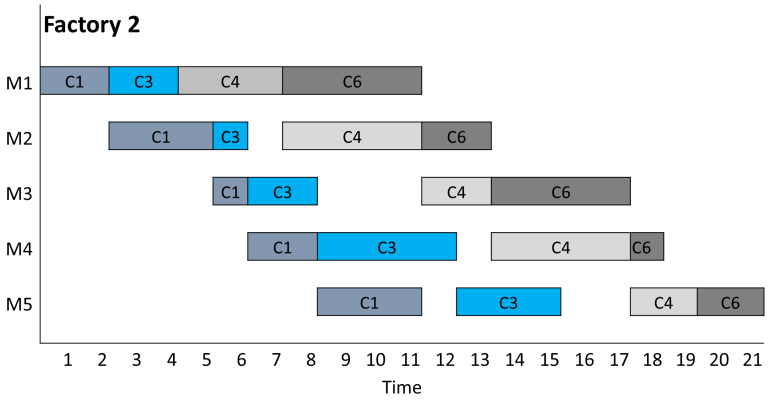
Assuming that we ignored the facility cost in fulfilling the jobs and considered scheduling the jobs at all factories as modelled in DPFSP. For instance, given that the solution of the DPFSP is $\{(1,8),(6),(4),(2,5),(3,7)\}$, i.e., customers 1 and 8 allocated to factory 1, customer 6 to factory 2, customer 4 to factory 3, customers 2 and 5 to factory 4 and customers 3 and 7 allocated to factory 5. Then, the objective (makespan) is calculated as follows: $f(x_{PFSP}) = C_{max} = max\{C_1, C_2, C_3, C_4, C_5\} = 16$. Note that, in the FLPPFSP example, the makespan $f(x_{PFSP})$ was 22, and this is attributed to the number of factories. Thus, with more factories, makespan will likely be reduced. However, if we consider the factory cost of fulfilling the



(a)



(b)



(c)

Figure 6.1: FLPPFSP - Example of Sub-PFSP Solutions at Selected Factories

jobs in DPFSP, this will cost 399 (i.e., 20+23+25+29+30+34+30+30+47+27+10+29+65). Thus, scheduling the jobs across all the factories is not cost-efficient. Invariably, if we further consider all makespan across the factories, the total makespan for FLPPFSP and DPFSP would be 51 (15+14+22) and 73 (15+13+15+14+16), respectively. Environmental sustainability has become an increasingly important issue for supply chains [246] because the associ-

ated social and environmental impacts are highly related to the operation and processing times of jobs [8]. Therefore, these calculations indicate that the FLPPFSP has the potential to achieve greater environmental and cost benefits for manufacturers than when DPFSP is considered independently.

In FLPPFSP, determining the optimal location of factories and obtaining the best scheduling plan across selected facilities are the two decisions that must be taken simultaneously. In tackling the FLPPFSP, we need to identify how the two problems (FLP and PFSP) are connected. There are several ways of connecting them depending on how they depend on each other and the choice of the number of objectives. The interdependence of the two problems in FLPPFSP have an implication on the algorithmic design for tackling the problem. A different problem presentation results to different efficiency of an optimisation method [42].

6.3.2 Problem Formulation

The FLPPFSP denotes P as defined from the formalism of the linked optimisation problem in Section 3.2.1 and we refer to the mathematical definition of FLPPFSP in Subsection 5.4.2 of Chapter 5 in Equation 5.34 and constrained by Contraints 5.2, 5.4, and 5.13 - 5.16. We establish a linked optimisation model of FLPPFSP based on facility cost and makespan.

Regarding FLPPFSP as seen in Figure 6.2, solution x_{FLP} of FLP determines the instance of PFSP and this modifies its fitness function everytime x_{FLP} interacts with it.

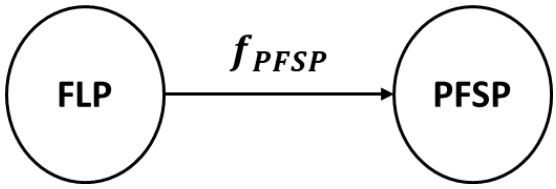


Figure 6.2: FLP and PFSP Linkages

6.3.3 Genetic Components for FLPPFSP

The sequential algorithmic approach considers two genetic algorithms, one for each problem in FLPPFSP. The two problems in FLPPFSP have two different solution representations, and that uniquely differentiates the two algorithms in terms of encoding and genetic operators used by each algorithm. In NSGALP and MCRGALP, we embed the different encodings and the genetic operators in a single algorithmic process.

Encoding

The FLPPFSP encoding uses binary-based encoding for FLP and permutation-based encoding for PFSP. The binary-based solution representation is required to address two issues;

factory selection and the assignment of jobs to selected factories. The permutation-based mechanism addresses the job sequence at each factory.

Given an FLP solution = {(00001), (10100), (00000), (01010)}, means that job 5 is processed in factory 1, jobs 1 and 3 are processed in factory 2 and jobs 2 and 4 are processed in factory 4. Note that factory 3 is not selected to process any jobs. In the permutation-based mechanism, job permutation occurs at the factories selected to process each job. The PFSP solution = {(5), (3,1), (2,4)} is a permutation of jobs with respect to the FLP solution. The representation of the linkages between the two solutions is driven by our linked optimisation problem framework provided in Section 3.2.1 of Chapter 3.

Initialisation

In each algorithmic approach, initialisation is carried out by randomly generating a population of size N. In terms of the sequential approach, the individual algorithms in Algorithm 7 generate their population separately and apply the genetic search process to the population. This is quite different in NSGALP and MCRGALP. In the initialisation process in Algorithms 8 and 9, each randomly generated solution of FLP is used to instantiate the PFSP and then, a random solution for the modified PFSP is generated and paired with the solution of the FLP.

Genetic Operators

A genetic method adopts crossover and mutation operators to improve solutions during a search process [248]. Here, we use the same pair of crossover and mutation in the individual approaches for the respective solution types. In terms of the crossover operators, half uniform crossover HUX operator is selected for the FLP solutions and partially mapped crossover PMX for PFSP solutions, respectively. Regarding mutation operators, we use BitFlip mutation for updating the FLP solutions and permutation swap mutation for the PFSP solutions, respectively. Partially mapped crossover and permutation swap mutation are widely used in the study of scheduling [246]. The genetic methodological framework uses the same crossover and mutation operators for all three approaches. The choice of the genetic operators used in the algorithmic approaches is driven by how they have been used in the literature for related problems.

In terms of HUX, uniform crossover has commonly been adopted as a genetic operator in several FLP studies ranging from single to multi-objective FLP [32] [166] [216]. It is a known fact that several variants of GAs has proven to be robust with respect to the parameter setting adopted in HUX. For instance, Konak et al. [216] consider a multi-objective FLP involving multiple competitors. They use a multi-objective to solve the problem using HUX as a crossover operator.

Similarly, several variants of GA applied to tackle FLP have adopted BitFlip mutation operator to diversify the offspring of the parent solutions [10] [166] [254] [218] [222] [352]

[103]. The choice of BitFlip (also known as Bitwise) mutation operator is influenced by its performance and how it has consistently been used for binary-coded GAs as seen in [103].

In the field of job scheduling, PMX crossover and permutation swap mutation operators have extensively been applied especially in the DPFSP/PFSP domain [246] [40]. This is because both operators are flexible according to the characteristics of DPFSP/PFSP [247] [369]. For instance, Huang [145] applied a hybrid GA using PMX crossover operator to improve the diversity of solutions of the DPFSP variant. Also, Bacha et al. [17] applied PMX operator to update solutions of the PFSP. In terms of permutation swap mutation, for example, Li and Chen [236] applied permutation swap mutation independently to every child obtained from the crossover process to improve solution diversity in the GA implementation proposed for the DPFSP.

In the sequential approach, in Algorithm 7, we adopt a binary-coded genetic algorithm A_{FLP} which uses HUX crossover and BitFlip mutation operators to generate offspring for the FLP. In terms of the PFSP, we use a permutation-coded genetic algorithm A_{PFSP} in the sequential approach. A_{PFSP} uses PMX and permutation swap mutation to update solutions. A tournament selection is employed in the sequential approach, which is used by algorithms A_{FLP} and A_{PFSP}. The tournament selection allows a selection of two parents to produce offspring [150].

The procedure for offspring generation is the same for Algorithms 8 and 9. The procedure is outlined as follows; Generate **n** offspring of FLP solutions from mating pool $\mathscr{R}^t_{FLP}$ using HUX crossover and BitFlip mutation operators. For each offspring generated for FLP, instantiate problem PFSP and randomly generate N solutions for PFSP. Next, perform crossover and mutation operations on N solutions of PFSP and generate **n** offspring and evaluate the N offspring and sort in descending order. Finally, select the best offspring from **n** offspring of PFSP and pair it with each offspring of FLP.

In addition, we use crowded binary tournament selection for NSGALP. The crowded binary tournament operator is a modified version of the binary tournament selector that incorporates ranking and diversity. In MCRGALP, we applied a multi-criteria algorithm (TOPSIS) to score each solution pair in a population incorporated in a tournament selection operator so that a minimum of two parent pairs can be selected for mating.

6.3.4 Experiments

A series of computational experiments are conducted to evaluate the performance of the proposed algorithmic approaches selected to tackle the linked problem. These experiments are conducted on the same computer environment with Intel Core i9, 2.4GHz, 32GB RAM, and Windows 10 Enterprise OS. The three algorithmic approaches are implemented in Java. This section is organised as follows. At first, the selected benchmark problems used in this experiment are introduced. Next, we pre-assessed the FLPPFSP to examine the degree of the linkages between the two sub-problems in FLPPFSP. Then, the selected performance

Table 6.3: Benchmark Problem Instances

Problem	**Benchmarks**	**Problem set**	**Problem size**	
FLP				**F**
	Beasley [24]	cap64, cap71	50	16
		cap94, cap101	50	25
		cap123, cap131	50	50
	Holmberg [175]	p1, p8, p9	50	10
		p13, p20, p21	50	20
		p50, p52, p54	100	10
		p51, p53, p55	100	20
		p56, p60, p64, p68	200	30
PFSP				**m**
	Taillard [343]	tai50_5_0 – tai50_5_9	50	5
		tai50_10_0 – tai50_10_9	50	10
		tai50_20_0 – tai_50_20_9	50	20
		tai100_5_0 – tai100_5_9	100	5
		tai100_10_0 – tai100_10_9	100	10
		tai100_20_0 – tai100_20_9	100	20
		tai200_10_0 – tai200_10_9	200	10
		tai200_20_0 – tai200_20_9	200	20

metric is given to measure the quality of the solutions generated by the competing algorithms. Next, the parameter settings employed in all the selected approaches are listed, and experimental results measured by the selected performance metric are presented and analysed. Finally, the performance of the selected approaches on the linked problem instances is presented.

Benchmark Problems

We evaluate the proposed algorithmic approaches on two sets of instances for both problems in FLPPFSP. The first set contains instances of the FLP, which are based on two benchmarks (Beasley [24], and Holmberg [175]). These instances are composed of 6 instances from Beasley and 16 instances from Holmberg. The second set contains instances of the PFSP, which are extracted from Taillard's benchmark. This is composed of 8 combinations of jobs by machines, and for each combination, there are 10 different instances. Each instance in the FLP benchmark is combined with each PFSP instance in terms of problem size. We assume that a customer corresponds to a job. So, we obtained 620 combined instances in total. See Table 6.4.

Table 6.4: Linked Problem Instances

FLP & PFSP Instances								
Problem Size	**No. of Machines**	**N0. of Factories**						**No. of Instances**
		10	**16**	**20**	**25**	**30**	**50**	
50	5	30	20	30	20	-	20	**120**
50	10	30	20	30	20	-	20	**120**
50	20	30	20	30	20	-	20	**120**
100	5	30	-	30	-	-	-	**60**
100	10	30	-	30	-	-	-	**60**
100	20	30	-	30	-	-	-	**60**
200	10	-	-	-	-	40	-	**40**
200	20	-	-	-	-	40	-	**40**
Total								**620**

Performance Metric

We use three performance metrics to assess the algorithmic approaches' behaviour for solving the linked optimisation problem. This includes; Hypervolume (HV) [405], Relative Hypervolume (RHV) and Multiplicative Epsilon [406]. These are widely-used unary quality multi-objective optimisation performance metrics [31] which we have adopted for the linked optimisation problem performance assessment.

Relative Hypervolume RHV

The relative hypervolume measures the proportion of hypervolume achieved by an individual approach. This is computed by dividing the hypervolume of approximations by individual approach by the hypervolume of the true Pareto front. A higher RHV indicates that approximations are closer to the true Pareto front.

$$RHV(\mathbf{Z}, A) = \frac{HV(A, \mathbf{r})}{HV(\mathbf{Z}, \mathbf{r})} \tag{6.9}$$

where $0 \leq RHV(\mathbf{Z}, A) \leq 1$

Hypervolume HV

HV considers the volume of the objective space dominated by an approximation set [404] bounded by a given reference point $\mathbf{r} \in \mathbb{R}^2$. HV provides a measure for both convergence and diversity, which is widely used as a comparison tool for multi-objective algorithms [252]. Here, the reference point $\mathbf{r}$ is the worst point used in the HV computation. In our experiments, all the HV values are normalised to (0,1). Higher HV values indicate a better performance of the corresponding approaches.

Multiplicative Epsilon ϵ

The Epsilon indicator gives a factor by which an approximation set is worse than another with respect to all objectives [406]. A lower Epsilon value corresponds to a better approximation set, regardless of the type of problem (minimisation, maximisation or mixed). We compute as follows;

$$epsilon(A,\mathbf{Z}) = \max_{\mathbf{z}\in\mathbf{Z}} \min_{a\in A} \max_{1\leq\iota\leq n} epsilon(a_\iota,\mathbf{z}_\iota) \tag{6.10}$$

where $epsilon(a_\iota,\mathbf{z}_\iota) = a_\iota/\mathbf{z}_\iota$

For each instance, we obtained two reference points from a combination of solution pairs generated by an algorithmic approach over multiple independent runs. The first reference point is a single point defined as r. We compute r by obtaining the maximum of each objective value for both problems from all the combined solution pairs and multiplying each value by a constant 1.5. The second reference point contains non-dominated points $\mathbf{Z}$ obtained from the combination solution pairs produced by individual algorithmic approaches over the entire independent runs per instance. $\mathbf{Z}$ is used as a Pareto-optimal for the computation of Relative Hypervolume and Epsilon metrics.

Parameter Settings

Table 6.5 shows the parameters used by the individual approach. To measure the behaviour of our selected algorithmic approaches for solving the FLPPFSP, we maintained the same parameter settings for the different genetic algorithms in all the approaches. We adopt the same termination criterion (i.e., the maximum number of fitness evaluations is set to 10000) among all algorithmic approaches. We use the same parameters for all approaches' respective crossover and mutation values. Each comparative algorithm was executed over 100 runs independently on each instance. An additional set of parameters are used by the TOPSIS (Technique for Order of Preference by Similarity to Ideal Solution) method adopted in the MCRGALP approach. We set each weight for the individual decision criterion to 0.5. This is to give equal importance to both problems so as to maintain balanced results.

In the parameter setting, fixed parameter values were used based on the consideration of the overall performance of the linked problem system. The parameter values are randomly selected from the range of values used for each problem in the literature [10] [32] [40] [145] [166] [179] [216] [246] [247] [254] [352] [369]. For individual problem in the linked problem, in isolation, several variations of parameters have been tested on problem instances in the literature. Evidence have shown that algorithmic performance is largely effected by quality of parameters selected. However, the nature of the linked problem is such that an optimal solution of one problem does not guarantee optimal overall solution. Therefore, optimising the parameters of an algorithmic process on one problem might have an adverse effect on the other problem.

Table 6.5: Parameter Settings

Parameters	**NSGALP**	**MCRGALP**	**SEQUENTIAL**	
No. of Algorithms	**1**	**1**	**2**	
No. of Independent Run	100	100	100	100
Population Size	100	100	100	100
Max Evaluations	10000	10000	10000	10000
Mating Pool Size	100	100	-	-
Offspring Population Size	100	100	-	-
HUXCrossover	0.9	0.9	0.9	-
PMXCrossover	0.1	0.1	-	0.1
BitFlipMutation	0.8	0.8	0.8	-
PermutationSwapMutation	0.5	0.5	-	0.5

6.3.5 Experimental Results and Analysis

Table 6.6: Mean values of relative hypervolume, hypervolume and epsilon metrics of MCRGALP, NSGALP and SEQ

Size	**F**	**m**	**Relative Hypervolume**			**Hypervolume**			**Epsilon**		
			MCRGALP	**NSGALP**	**SEQ**	**MCRGALP**	**NSGALP**	**SEQ**	**MCRGALP**	**NSGALP**	**SEQ**
50	10	5	0.8454	**0.9559**	0.7381	0.2380	**0.2693**	0.2080	1.1344	**1.0737**	1.5233
50	10	10	0.8396	**0.9506**	0.7605	0.2190	**0.2481**	0.1985	1.1288	**1.0660**	1.3883
50	10	20	0.8457	**0.9589**	0.7969	0.2016	**0.2288**	0.1902	1.1058	**1.0575**	1.2646
50	16	5	0.9213	**0.9622**	0.8237	0.2982	**0.3128**	0.2620	1.0773	**1.0692**	1.8271
50	16	10	0.9159	**0.9580**	0.8282	0.2786	**0.2928**	0.2474	1.0726	**1.0639**	1.5983
50	16	20	0.9144	**0.9574**	0.8364	0.2612	**0.2749**	0.2348	1.0664	**1.0485**	1.4363
50	20	5	0.8317	**0.9371**	0.8155	0.3263	**0.3678**	0.3192	1.1984	**1.0922**	1.4377
50	20	10	0.8404	**0.9361**	0.8300	0.3005	**0.3349**	0.2961	1.1629	**1.0831**	1.3005
50	20	20	0.8497	**0.9419**	0.8489	0.2728	**0.3026**	0.2718	1.1310	**1.0689**	1.1956
50	25	5	0.8529	**0.9586**	0.7266	0.3551	**0.4003**	0.3014	1.1807	**1.0701**	2.5982
50	25	10	0.8870	**0.9563**	0.7995	0.3185	**0.3453**	0.2848	1.1196	**1.0608**	1.7053
50	25	20	0.8779	**0.9575**	0.7908	0.3021	**0.3315**	0.2698	1.1145	**1.0592**	1.5688
50	50	5	0.8613	**0.9553**	0.8006	0.3452	**0.3839**	0.3212	1.1506	**1.0704**	1.7663
50	50	10	0.8691	**0.9569**	0.8208	0.3084	**0.3406**	0.2911	1.1225	**1.0578**	1.4398
50	50	20	0.8703	**0.9571**	0.8299	0.2956	**0.3259**	0.2816	1.1191	**1.0546**	1.3180
100	10	5	0.8470	**0.9186**	0.7896	0.2655	**0.2879**	0.2472	1.1544	**1.1116**	1.3033
100	10	10	0.8461	**0.9164**	0.8115	0.2547	**0.2732**	0.2414	1.1425	**1.1077**	1.2456
100	10	20	0.8528	**0.9225**	0.8294	0.2332	**0.2522**	0.2265	1.1181	**1.0876**	1.1873
100	20	5	0.7869	**0.9169**	0.6988	0.3255	**0.3789**	0.2893	1.2822	**1.1293**	1.6547
100	20	10	0.7815	**0.9145**	0.7218	0.3067	**0.3588**	0.2833	1.2724	**1.1210**	1.5217
100	20	20	0.7875	**0.9157**	0.7390	0.2834	**0.3294**	0.2659	1.2426	**1.1200**	1.3849
200	30	10	0.8539	**0.8857**	0.8405	0.2229	**0.2312**	0.2194	1.1171	**1.1111**	1.2264
200	30	20	0.8557	**0.8864**	0.8513	0.2062	**0.2136**	0.2052	1.0989	**1.0822**	1.1695

Performance Metrics

The mean results of the three metrics are listed in Table 6.6, where the best values are highlighted in bold font. It can be observed from the results in the table that NSGALP shows the best performance for the three metrics. However, the mean results of the three algorithmic approaches appear close to each other, especially in the hypervolume metric. NSGALP significantly outperforms the other competing algorithmic approaches, specifically in the relative hypervolume and epsilon metrics. We conducted a Wilcoxon signed-rank test to check the statistical significance of the differences in the empirical results. Table 6.7 sum-

marises the corresponding p values among the compared algorithms on instances grouped by problem size, factories and machines. The p values show significant differences among the algorithmic approaches, mostly in some groups of correlated instances. However, we also observed that 21.7% of the problem instances indicate no significant difference in performance. These are highlighted in bold font in Table 6.7. The highlighted values in bold text refer to values above 0.05, indicating that there is no significant difference between the performance of a pair of algorithmic methods on problem instances. This suggests that MCRGALP and sequential approaches efficiently tackle some instances of the linked problem, but that depends on how the two problems are linked. However, the selection of the best approach points toward NSGALP; there are several explanations for why NSGALP outperforms the other two using these metrics. One reason is that as performance metrics are multi-objective based, they are largely influenced by the number of non-dominated points produced by an algorithm. Unlike the Pareto set produced by NSGALP, the two other approaches produce a single-point solution pair, reducing their chances of obtaining a higher metric score. This makes it difficult to use appropriate performance metrics in comparing all the algorithmic approaches. Thus, the metrics are more biased towards NSGALP. In addition, the Pareto set obtained by NSGALP includes extreme points, which, to a large extent, are a mix of trade-offs. Including these points in the performance metrics computation will improve each metric score.

Table 6.7: The p values of all metrics among the three algorithmic approaches on different problem combinations

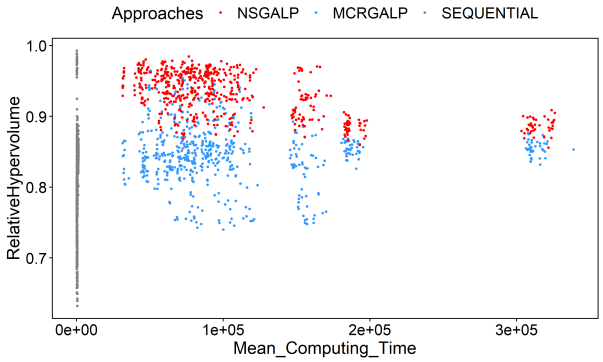
Size	**F**	**m**	**Epsilon Metric**			**Hypervolume Metric**			**Relative Hypervolume**		
			MCRGALP vs SEQ	**NSGALP vs MCRGALP**	**NSGALP vs SEQ**	**MCRGALP vs SEQ**	**NSGALP vs MCRGALP**	**NSGALP vs SEQ**	**MCRGALP vs SEQ**	**NSGALP vs MCRGALP**	**NSGALP vs SEQ**
50	10	5	6.51E-11	3.98E-07	6.51E-11	2.58E-09	1.20E-09	6.51E-11	6.51E-11	6.51E-11	6.51E-11
50	10	10	6.51E-11	4.99E-09	6.50E-11	9.86E-10	8.01E-11	6.51E-11	9.84E-11	6.51E-11	6.51E-11
50	10	20	3.02E-11	2.61E-10	3.02E-11	5.46E-06	6.07E-11	4.98E-11	4.57E-09	3.02E-11	3.02E-11
50	16	5	**8.50E-02**	**2.31E-01**	**5.77E-02**	7.08E-04	**1.22E-01**	5.71E-04	**6.83E-01**	1.55E-05	**1.29E-01**
50	16	10	**6.08E-01**	**1.24E-01**	**3.30E-01**	**7.27E-02**	**1.51E-01**	**5.78E-02**	**7.58E-01**	1.48E-04	**1.82E-01**
50	16	20	**6.05E-01**	6.51E-03	**3.52E-01**	2.52E-02	**1.48E-01**	1.92E-02	**7.83E-01**	1.95E-05	**2.15E-01**
50	20	5	3.02E-11	3.02E-11	3.02E-11	**7.39E-01**	5.61E-05	2.44E-09	4.21E-02	3.02E-11	3.02E-11
50	20	10	2.87E-10	3.02E-11	3.02E-11	**9.47E-01**	1.17E-04	8.84E-07	**2.97E-01**	3.02E-11	3.02E-11
50	20	20	1.03E-06	3.02E-11	3.02E-11	**4.12E-01**	1.11E-04	2.15E-06	**3.63E-01**	3.02E-11	3.02E-11
50	25	5	2.17E-03	2.17E-03	2.17E-03	1.52E-02	2.15E-02	3.29E-03	2.15E-02	2.17E-03	2.17E-03
50	25	10	1.69E-06	1.32E-04	7.05E-07	2.17E-03	**9.83E-02**	1.14E-04	7.22E-03	1.69E-06	7.05E-07
50	25	20	7.48E-06	2.62E-04	3.39E-06	2.51E-02	2.51E-02	8.97E-03	1.61E-02	3.39E-06	3.39E-06
50	50	5	1.48E-07	9.36E-07	1.48E-07	4.40E-02	6.06E-03	8.10E-05	3.97E-06	1.48E-07	1.48E-07
50	50	10	3.39E-06	3.36E-05	3.39E-06	9.71E-02	3.81E-02	4.21E-03	6.71E-04	3.39E-06	3.39E-06
50	50	20	2.14E-05	2.14E-05	7.47E-06	**1.75E-01**	3.66E-02	8.24E-03	2.25E-03	7.47E-06	7.47E-06
100	10	5	1.07E-09	2.05E-03	1.33E-10	1.78E-04	1.04E-04	4.57E-09	7.04E-07	2.37E-10	3.69E-11
100	10	10	1.94E-09	8.99E-03	1.20E-09	2.32E-02	7.82E-04	4.32E-07	9.25E-04	1.48E-10	8.88E-11
100	10	20	3.35E-08	5.87E-04	6.12E-10	8.77E-02	1.04E-04	2.38E-07	5.32E-03	4.98E-11	2.61E-10
100	20	5	3.02E-11	3.02E-11	3.02E-11	6.74E-06	6.72E-10	1.21E-10	3.01E-07	3.02E-11	3.02E-11
100	20	10	6.72E-10	3.02E-11	3.02E-11	1.17E-05	1.33E-10	1.33E-10	6.28E-06	3.02E-11	3.02E-11
100	20	20	2.78E-07	3.02E-11	3.02E-11	2.25E-04	5.57E-10	9.92E-11	8.66E-05	3.02E-11	3.02E-11
200	30	10	1.44E-14	**1.96E-01**	1.55E-14	7.27E-02	6.38E-05	9.45E-07	4.06E-05	5.48E-14	3.03E-14
200	30	20	1.44E-14	1.07E-05	1.55E-14	**4.44E-01**	1.47E-04	1.17E-05	4.91E-02	2.18E-13	4.78E-13

Computational Time

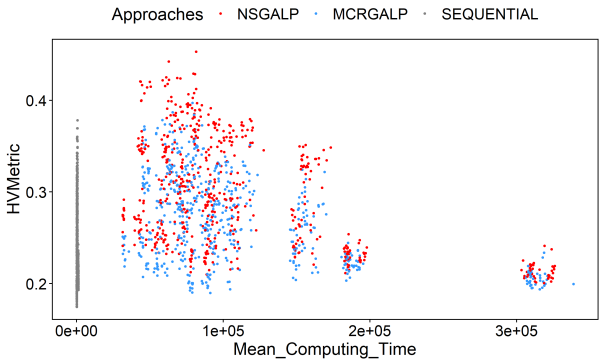
We also consider the performance of the algorithmic approaches based on computational time in milliseconds. Figure 6.3 shows three plots of mean computational time against the performance metrics we have used in assessing the performance of the three algorithms. Figure 6.3a shows the mean computing time against the relative hypervolume metric. Figure 6.3b shows the mean computing time against the hypervolume metric, and Figure 6.3c shows the mean computing time against the epsilon metric. From the three plots, the grey points represent the mean computing time achieved by the sequential approach, the blue points represent the mean computing time achieved by MCRGALP, and the red points represent the ones achieved by NSGALP. There is no doubt that the sequential approach required less computational time than the other approaches in all combinations of problem instances. It is also interesting to see that, unlike the sequential approach, the computing time required by NSGALP and MCRGALP increases as problem size increases. This can be seen in the partitions shown by NSGALP and MCRGALP on the three plots. This also suggests that the sequential approach is robust with different problem sizes in terms of computational time. However, the three performance metrics suggest contrary views in most problem instances.

Correlation Analysis

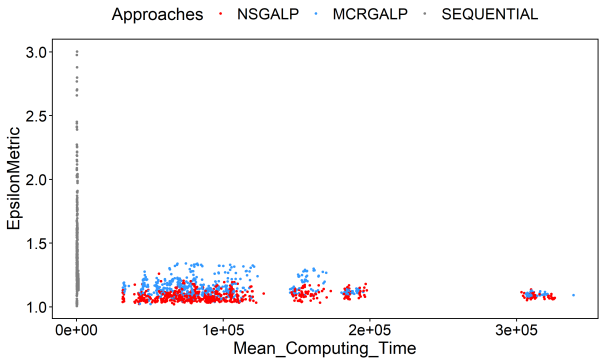
We further consider algorithm performance in terms of the correlation score obtained by randomly generated solutions as discussed in Section **??**. We compare the performance of the competing algorithms on instances that obtained the lowest, median and highest correlation coefficients. See correlation scores in Table 6.8. The essence is to accept the hypothesis that problem instances of FLPPFSP which are uncorrelated or are weakly and positively correlated could be tackled using simple approaches like sequential methods to achieve a joint optimal solution in less computational time. Figure 6.4 shows the Pareto approximation set obtained by the competing algorithmic approaches on the instances with respective minimum, median and maximum correlation scores. The Pareto approximations are achieved over 100 independent experimental runs. Figure 6.4a. shows the Pareto approximation sets on problem instance with the lowest correlation score (-0.26). This problem instance gives an indirect relationship between the two problems in FLPPFSP. This means that, as we try to minimise the FLP cost, this consequently increases the PFSP scheduling's makespan. There is a high level of trade-off of makespan for factory and distribution cost reduction. Using a sequential method in this problem scenario as seen in Figure 6.4a, would obviously optimise the FLP problem and sub optimise the PFSP problem. Thus, other methods might be more efficient in tackling such problem scenarios. For instance, MCRGALP tries to achieve a solution set that collective considers the two minimisation problems, although both problems are not fully optimised but are better than sub-optimised points. Figure 6.4b shows how the sequential approach produces the best Pareto approximation



(a)



(b)



(c)

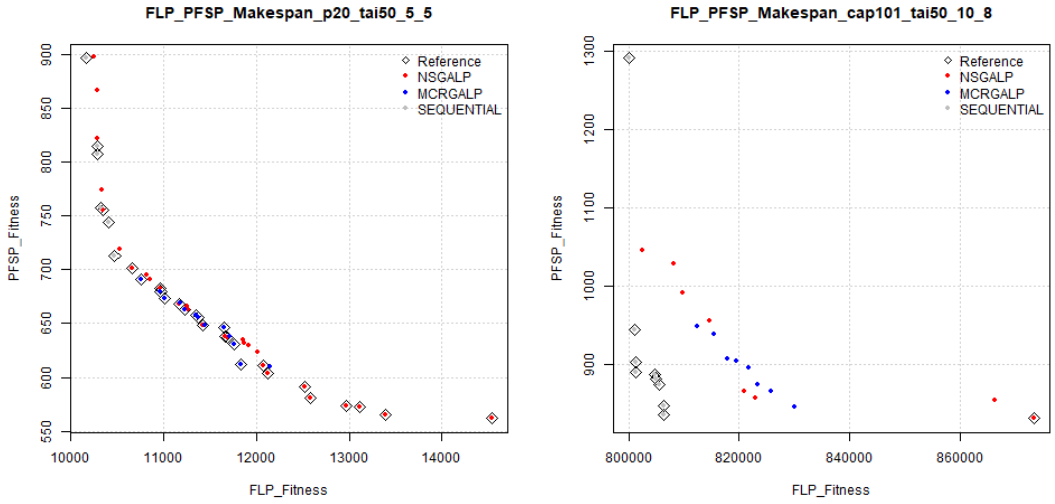
Figure 6.3: Mean Computation Time against Performance Metrics

set. This is largely possible because the problem instance indicates a positive and weak correlation between the two problems in FLPPFSP. Thus, solving the FLP problem in such a problem instance scenario would have minimal impact on obtaining an optimal outcome for the PFSP problem. This suggests that a sequential approach is more effective and efficient with uncorrelated or weak and positively correlated problem instances of FLPPFSP. Figure 6.4c shows the Pareto approximation sets obtained by the three approaches on a problem instance with a correlation score of 0.86. The sequential approach is biased towards optimising the first problem and then producing sub-optimised solutions for the sec-

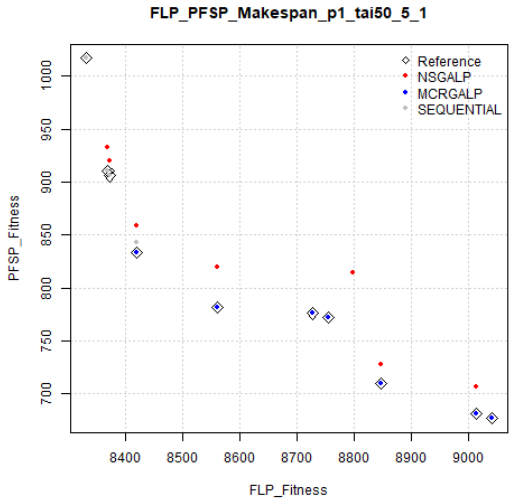
ond problem.

Table 6.8: Problem instances with minimum, median and maximum correlation coefficients

Problem size	F	m	Correlation Score
50	20	5	-0.26
50	25	10	0.4928
50	10	5	0.86



(a) (b)

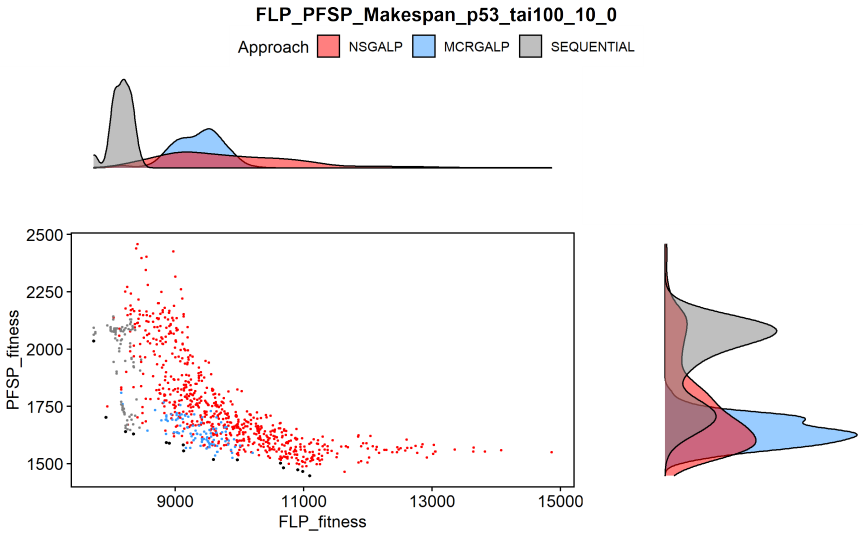


(c)

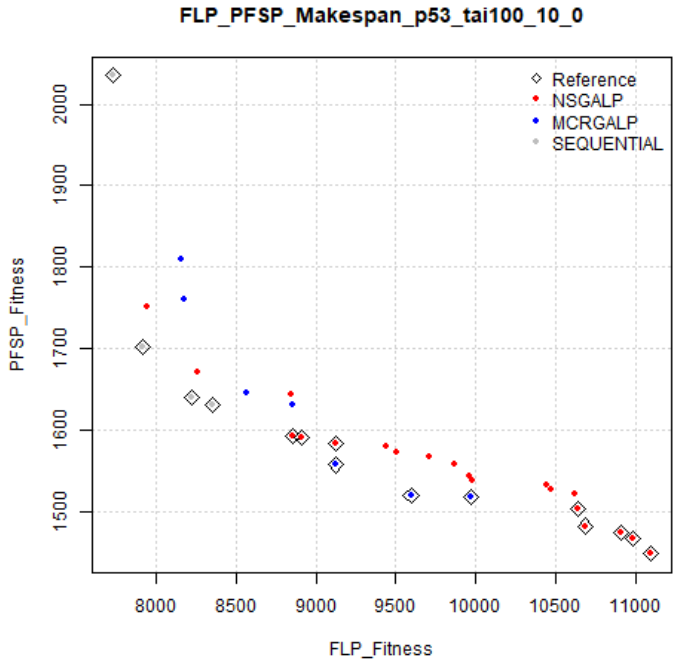
Figure 6.4: Examples of Pareto Fronts by all Algorithmic Approaches

Supply Chain Perspective

Considering the two problems from the perspective of two companies involved in a supply chain, our results can offer guidance on the benefits and costs they are likely to experience based on the approach used to solve the overall problem. For example, in Figure 6.5b, in an attempt to minimise the factory and distribution cost associated with the FLP problem, the sequential method sacrifices the makespan associated with the PFSP problem. As shown by the distributions in Figure 6.5a, the sequential method tends to prefer the first problem (FLP) as there is a strong peak at a low FLP fitness value in contrast to the PFSP, where there is a strong peak with a middle value below the other two methods. The NSGALP is quite interesting due to the large extent of variability in the fitness values of the FLP criterion. This can be seen in the distribution of the NSGALP as its peak is lower than the other two methods. The NSGALP attempts to solve the first problem with a view to trade off the other problem. However, trading off the second problem was not that much as it can be seen that there is a peak at the lower PFSP fitness value. However, the MCRGALP achieved a sweet spot for 99% of problem instances, but this does not quantify in terms of the performance metrics used. Therefore, in deciding how to solve the problem with the sequential and NSGALP, it is more apparent that both companies must consider the impact that optimising one problem will have on the other and decide if the resulting costs/benefits are acceptable. In comparison, the MCRGALP tends to maintain a balanced compromise on both problems.



(a)



(b)

Figure 6.5: Distribution of solutions found by all algorithmic approaches over 100 independent runs on problem size 100 with F=20 and m=10.

6.4 Case Study 2: JAP (p_1) and TSP (p_2)

JAP and TSP are two distinct classical optimisation problems whose integration is applicable in hospital resource planning and field service management [375] as well as logistics and supply chains. The integration of JAP and TSP can be applied to the health visiting problem, which can provide sufficient capacity and allow spare capacity to be redeployed

to respond to caseloads of COVID. For example, in the first wave of COVID-19 in England, three of five mandated health visiting services were paused to redeploy health visitors to respond to caseloads across several communities [91]. It was then recommended that a clear plan for health visiting service is required to ensure sufficient capacity and manage missed appointments backlog [91].

This section investigates the integration of JAP and TSP using a linked optimisation framework to minimise the total cost of jobs performed by service personnel/agents and the total travelling cost of visiting the assigned job locations by the agents. We call this linked problem the Job Assignment Problem and Travelling Salesman Problem (JAPTSP).

We believe that interacting optimisation problems are connected in diverse ways and may require different and complex algorithmic designs to tackle them. This case study considers 114 combined problem instances of existing benchmarks in JAP and TSP and employed the proposed algorithmic approaches in Section 6.2 on the combined problem instances. These algorithms include; Nondominated Sorting Genetic Algorithm for Linked Problem (NSGALP), Multi-Criteria Ranking Genetic Algorithm for Linked Problem (MCRGALP), and Sequential approach. Performance comparison of the three approaches was assessed on four performance metrics, including Relative hypervolume, hypervolume, inverted generational distance, and Epsilon. We also consider other factors like computational time, correlation analysis and service/supply chain perspective.

6.4.1 Problem Background

JAPTSP refers to a class of optimisation problem where service personnel/agents are assigned to perform tasks in different cities. JAPTSP is an extension of the multiple travelling salesman problems (MTSP) and workforce scheduling and routing problem studied in the literature. Different real-life problems can be modelled using the JAPTSP framework.

So far, different variations of JAPTSP have been explored in the literature, and several approaches have been proposed for tackling them [62]. A prominent study is discussed in a survey undertaken by [62] in the context of workforce scheduling. They refer to scenarios where personnel carry out tasks at different locations, such as Workforce Scheduling and Routing Problem (WSRP). In the study of WSRP, [375] describes an iterated local search ILS algorithm. The paper evaluated ILS against a mixed integer programming (MIP) model and an adaptive larger neighbourhood search (ALNS) algorithm. Similarly, [63] proposed a greedy heuristic algorithmic design for five time-dependent constraints for WSRP.

Other variations of JAPTSP can be seen in [56] involving the investigation of a Travelling Maintainer Problem (TMP) based on a generalised formulation of TSP. Their proposed problem seeks to find the best route for maintainers that minimises the travel, maintenance, and expected failure cost for all cities. The authors present genetic algorithm and particle swarm optimisation solutions for comparison in the TMP study. Similarly, [398] adopts a genetic algorithm for a team scheduling problem. The authors consider a photographic

studio where multiple teams are scheduled to different secondary schools in the scheduling problem. The team scheduling problem involves multiple TSPs, which consider total distance travelled and time consumed at the location into a single cost function for overall optimality. Also, [13] presents a mixed integer programming for multi-depot multiple travelling salesman problems (MmTSP) where an individual salesman travels from a particular location to a set of locations to complete tasks and return to the original location. [336] applied NSGA-II framework to tackle MTSP as a multi-objective problem. They designed some novel representation, crossover and mutation operators to improve search behaviours so that the total travel distance and the range between all salesmen can be minimised.

In tackling the JAPTSP, we need to identify how the two problems (JAP and TSP) are connected. There are several ways of connecting them depending on how they interdepend and the number of objectives chosen. In JAPTSP, determining the optimal job assignment and obtaining the best multiple permutations of tours are the two decisions that must be taken simultaneously. Integrating the two problems in JAPTSP causes complexity in designing appropriate algorithms for solving the problem. A different presentation of a problem may result in different efficiency of the optimisation method in solving the problem [42].

6.4.2 Problem Formulation

In JAPTSP, we seek to minimise the cost of performing a set of jobs assigned to a set of agents and minimise the travel distance of visiting the assigned job locations. In the problem, there are **n** cities with given distance matrix $\{d_{jk}\}$. Also, there are **n** jobs to be assigned to m agents/personnel with given availability/skill capacity/requirements. Here, the number of jobs for JAP corresponds to the number of locations in TSP. Each job in the JAP has a location in TSP. Given that the solution of the JAP is $x_{JAP} = (x_{ij})_{j=1}^{\mathbf{n}}$. x_{JAP} represents a solution of JAP containing $x_{ij} \in x_{JAP}$ and x_{ij} denotes agent i assigned to job j in solution x_{JAP}. This creates multiple TSP problems, each TSP per agent. Assuming m agents are scheduled for **n** jobs, then, TSP = $\{TSP_1, TSP_2, \cdots, TSP_m\}$. TSP is therefore minimised as $f^2_{x_{JAP}}(\mathbf{x}_{TSP}) = \sum_{i=1}^{m} f^2(TSP_i)$.

The mathematical formulation of the objective function for JAPTSP is presented in Equation 5.35 in Chapter 5. The problem is subject to contraints 5.9 - 5.11 in Chapter 5.

6.4.3 Genetic Components for JAPTSP

In the sequential algorithmic approach, two genetic algorithms are used, one for each problem in JAPTSP. The two problems in JAPTSP have two different solution representations, and that uniquely differentiates the two algorithms in terms of encoding and genetic operators used by each algorithm. In NSGALP and MCRGALP, we embed the different encodings and the genetic operators in a single algorithmic process.

Encoding

The encoding of JAPTSP uses two mechanisms; integer-based encoding for JAP and permutation based encoding for TSP. The Integer-based solution representation addresses the assignment of jobs to agents. The permutation-based mechanism addresses a sequence of travel by agents. For instance, assuming an instance of JAPTSP is to assign 6 jobs to 2 agents and given that the solution to the JAP is $x_{JAP} = 121221$. This means that jobs 1, 3 and 6 are assigned to agent 1 and jobs 2, 4 and 5 are assigned to agent 2. The solution creates two TSP problems, TSP_1 and TSP_2. Figure 6.6 shows the encoding structure of the JAPTSP linked problem. The solution to the JAP instantiates the TSP, and that creates sub-TSPs. This allows permutation at each subset of the TSP.

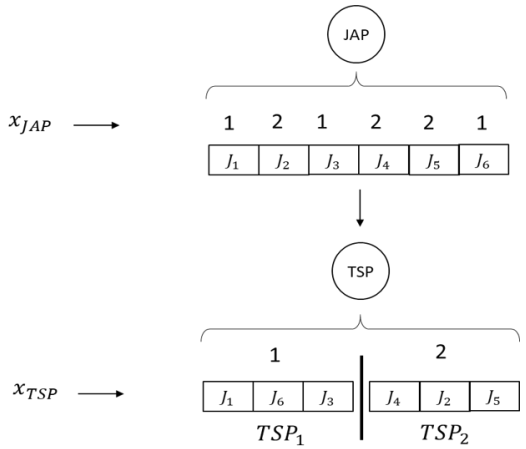


Figure 6.6: JAPTSP Encoding Example

Initialisation

The initialisation process for each algorithmic approach is the same as detailed in Section 6.3. Initialisation is carried out by randomly generating a population of size N.

Genetic Operators

Here, we use the same pair of crossover and mutation in the individual approach for the respective solution types. We use the Integer SBX crossover operator for the JAP solutions and partially mapped crossover PMX for TSP solutions. In terms of mutation operators, we use Integer Polynomial mutation for updating the JAP solutions and permutation swap mutation for the TSP solutions, respectively. The genetic methodological framework uses the same crossover and mutation operators for all three approaches.

In the study of generalised assignment problem, integer-based simulated binary crossover (SBX crossover) and integer based polynomial mutation are widely used operators for effectively obtaining optimal solutions [178]. These operators have been used as search strategy in several algorithms (i.e., improved multi-objective particle swarm optimisation algo-

rithm (IMPOSE) [217], and enhanced non-dominated sorting genetic algorithm II (ENSGA-II) [195]) to facilitate the sharing of elitist information among external archive [217]. These operators are adopted to improve the search performance of the algorithmic approaches used in this chapter. The choice of these operators was influenced by the work of Hu et al. [178] who adopted SBX crossover and polynomial mutation operators as search strategy after comparing different methods of crossover and mutation operators for container storage space assignment problem. In terms of the operators adopted for the TSP, we consider the structure and settings for the PFSP algorithmic adaptation in Section 6.3 (i.e., PMX crossover and permutation swap mutation operators). The PMX crossover and permutation swap mutation operators are widely used standard operators for permutation-based problems like TSPs [55].

In the sequential approach, in Algorithm 7, we adopt an integer-coded genetic algorithm A_{JAP} which uses integer SBX crossover and integer polynomial mutation operators to generate offspring for the JAP. In terms of the TSP, we use a permutation-coded genetic algorithm A_{TSP} in the sequential approach. A_{TSP} uses PMX and permutation swap mutation to update solutions. The two algorithms (A_{JAP} and A_{TSP}) in the sequential approach use the tournament selection operator. Tournament selection allows a minimum of two parent selection for offspring reproduction [150].

The procedure for offspring generation is the same for Algorithms 8 and 9. The procedure is outlined as follows;

- Generate **n** offspring of JAP solutions from mating pool $\mathscr{R}^t_{JAP}$ using integer SBX crossover and integer polynomial mutation operators.
- For each offspring generated for JAP, instantiate problem TSP and randomly generate N solutions for TSP.
- Perform crossover and mutation operations on N solutions of TSP and generate **n** offspring
- Evaluate the N offspring and sort in descending order
- Select best offspring from **n** offspring of TSP and pair with each offspring of JAP

In addition, we use crowded binary tournament selection for NSGALP. The crowded binary tournament operator is a modified version of the binary tournament selector that incorporates ranking and diversity. In MCRGALP, we applied a multi-criteria algorithm (TOPSIS) to score each solution pair in a population incorporated in a tournament selection operator so that a minimum of two parent pairs can be selected for mating.

Table 6.9: Linked Problem Instances

Problem Size	**No. of Agents**	**No. of Instances**
100	5	**30**
100	10	**30**
100	20	**24**
200	5	**10**
200	10	**10**
200	20	**10**
Total		**114**

6.4.4 Experiments

We performed a series of computational experiments to evaluate the performance of the proposed algorithmic approaches selected to tackle the linked problem. We use the same computer environment to conduct the experiments with Intel Core i9, 2.4GHz, 32GB RAM, and Windows 10 Enterprise OS. The three algorithmic approaches are implemented in Java.

Benchmark Problems

We evaluate the proposed algorithmic approaches on two sets of instances for both problems in JAPTSP. The first set is instances of the JAP, which are based on Beasley [24] benchmark. This is composed of a combination of 29 instances with a mix of job and agent sizes. The second set contains TSP instances extracted from Gerhard's [313] benchmark. This is composed of 8 instances of TSP. Each instance in JAP benchmark is combined with each TSP instance based on the corresponding problem size. We assume that a job in JAP corresponds to a city in TSP. So, we obtained 114 combined instances in total. See Table 6.9.

Performance Metric

We included Inverted Generational Distance (IGD) [31] in addition to the three performance metrics (i.e., Hypervolume (HV) [405], Relative Hypervolume (RHV), and Multiplicative Epsilon [406]) used in Section 6.3. We refer to Section 6.3 for details of the three metrics.

Inverted Generational Distance IGD

IGD assesses the quality of approximations achieved by a multi-objective algorithm to the Pareto front [31]. The metric measures how the approximations converge towards the true Pareto front. The smaller the IGD value, the closer the calculated front to the true Pareto front [252]. IGD is calculated as follows:

$$IGD(A,\mathbf{Z}) = \Big(\frac{1}{|\mathbf{Z}|}\sum_{i=1}^{|\mathbf{Z}|}\min_{a\in A}\mathbf{d}(\mathbf{z},a)^2\Big)^{\frac{1}{2}} \tag{6.11}$$

where $\mathbf{d}(\mathbf{z}, a) = \sqrt{\sum_{\iota}^{n}(\mathbf{z}_{\iota}, a_{\iota})}$ with a_{ι} being the ιth fitness value of point a from the approximations A and $\mathbf{z}_{\iota}$ being an ιth fitness value of point $\mathbf{z}$ from the true Pareto front $\mathbf{Z}$.

Empirical Attainment Function EAF

In addition to the selected performance metrics, we utilise an empirical comparison method that explicitly describes the probabilistic distribution of the outcomes obtained by the individual algorithmic approach over multiple independent optimisation runs [137].

Parameter Settings

Table 6.10 shows the parameters used by the individual approach. To measure the behaviour of our approaches for solving the JAPTSP, we maintained the same parameter settings for the different genetic algorithms in all the approaches. We adopt the same termination criterion (i.e., the maximum number of fitness evaluations is set to 10000) among all algorithmic approaches. We use the same parameters for all approaches' respective crossover and mutation values. Each comparative algorithm was executed over 100 independent runs on each combined instance. An additional set of parameters are used by the TOPSIS (Technique for Order of Preference by Similarity to Ideal Solution) method adopted in the MCRGALP approach. We set the weight for JAP fitness to 0.35 and its constraint, in case of violation, to 0.30. The weight for TSP is set to 0.35. The implication for these parameter values is that the two problems have equal importance and are considered when making a choice on the best solution pair. We use fixed parameter values by randomly selecting each value from range of values considered in the literature. For instance, we randomly selected 0.9 for the integer SBX crossover from the range of values used in Hu et al. [178]. The rationale for this approach is driven by the nature of the linked problem because using optimal parameter values for one problem does not guarantee an optimal overall solution. In terms of the TSP, as a permutation-based problem, the chapter considers the structure and settings for the PFSP algorithmic adaptation in Section 6.3.

Table 6.10: Parameter Settings

Parameters	**NSGALP**	**MCRGALP**	**SEQUENTIAL**	
No. of Algorithms	**1**	**1**	**2**	
Experimental Runs	100	100	100	100
Population Size	100	100	100	100
Max Evaluations	10000	10000	10000	10000
Mating Pool Size	100	100	-	-
Offspring Size	100	100	20	20
IntegerSBXCrossover	0.9	0.9	0.9	-
PMXCrossover	0.1	0.1	-	0.1
Integer Polynomial Mutation	1	1	1	1
Permutation Swap Mutation	0.5	0.5	-	0.5

Table 6.11: Mean values of relative hypervolume, hypervolume, inverted generational distance and Epsilon metrics of MCRGALP, NSGALP and SEQUENTIAL

Size	**m**		**RHVMetric**	
		MCRGALP	**NSGALP**	**SEQUENTIAL**
100	5	0.749	**0.775**	0.703
100	10	0.711	**0.778**	0.670
100	20	0.670	**0.763**	0.610
200	5	0.794	**0.834**	0.794
200	10	0.734	**0.815**	0.758
200	20	0.697	**0.808**	0.721
Size	**m**		**HVMetric**	
		MCRGALP	**NSGALP**	**SEQUENTIAL**
100	5	0.174	**0.180**	0.163
100	10	0.186	**0.204**	0.172
100	20	0.195	**0.224**	0.171
200	5	0.154	**0.163**	0.154
200	10	0.157	**0.175**	0.161
200	20	0.159	**0.186**	0.163
Size	**m**		**IGDMetric**	
		MCRGALP	**NSGALP**	**SEQUENTIAL**
100	5	7048.24	**6544.26**	10598.16
100	10	8716.75	**7049.15**	12260.79
100	20	12505.42	**8987.10**	17753.56
200	5	11026.66	**7454.52**	13790.09
200	10	14559.31	**9678.85**	13654.81
200	20	18550.02	**12371.31**	17784.36
Size	**m**		**EpsilonMetric**	
		MCRGALP	**NSGALP**	**SEQUENTIAL**
100	5	**1.186**	1.187	1.226
100	10	1.237	**1.195**	1.280
100	20	1.301	**1.188**	1.448
200	5	1.134	**1.101**	1.125
200	10	1.195	**1.133**	1.177
200	20	1.236	**1.140**	1.217

6.4.5 Experimental Results and Analysis

Performance Metrics

Table 6.11 shows the mean results of the four metrics. The best values are highlighted in bold font. NSGALP shows the best performance across all the four metrics, although the mean results appear to be close to each other, most especially in the hypervolume metric. MCRGALP slightly outperforms NSGALP and sequential approaches in problem instance ($Size = 100$ and $m = 5$) in terms of epsilon metric. To check whether the difference between the statistical results is significant or not, a Wilcoxon signed-rank test at 0.05 significance level is conducted on these results. Table 6.12 summarises the corresponding p values

Table 6.12: The p values of all metrics among the three algorithmic approaches on different problem combinations

Size	**m**		**RHV**	
		NSGALP Vs SEQ	**NSGALP Vs MCRGALP**	**MCRGALP Vs SEQ**
100	5	1.49E-04	**1.30E-01**	9.88E-03
100	10	4.80E-07	3.82E-10	**8.24E-02**
100	20	1.20E-06	1.34E-08	3.28E-02
200	5	**8.90E-02**	**7.57E-02**	**9.70E-01**
200	10	4.52E-02	3.30E-04	**5.71E-01**
200	20	3.76E-02	5.83E-04	**5.71E-01**
Size	**m**		**HV**	
		NSGALP Vs SEQ	**NSGALP Vs MCRGALP**	**MCRGALP Vs SEQ**
100	5	3.03E-03	**3.40E-01**	1.11E-04
100	10	2.92E-02	6.38E-03	2.39E-04
100	20	**5.39E-02**	3.53E-03	7.21E-05
200	5	4.52E-02	**1.86E-01**	1.01E-03
200	10	**6.40E-02**	**1.40E-01**	1.71E-03
200	20	**1.21E-01**	2.57E-02	7.28E-03
Size	**m**		**IGD**	
		NSGALP Vs SEQ	**NSGALP Vs MCRGALP**	**MCRGALP Vs SEQ**
100	5	3.50E-03	**2.90E-01**	1.22E-02
100	10	5.08E-03	**5.01E-02**	**1.67E-01**
100	20	1.01E-03	6.29E-03	**1.40E-01**
200	5	2.57E-02	**1.40E-01**	**8.90E-02**
200	10	**1.40E-01**	2.11E-02	**2.41E-01**
200	20	**7.57E-02**	7.28E-03	**2.41E-01**
Size	**m**		**Epsilon**	
		NSGALP Vs SEQ	**NSGALP Vs MCRGALP**	**MCRGALP Vs SEQ**
100	5	2.01E-04	**9.00E-01**	6.36E-05
100	10	2.84E-04	**3.55E-01**	1.39E-06
100	20	**8.45E-01**	4.88E-03	3.75E-04
200	5	2.57E-02	**2.73E-01**	4.40E-04
200	10	5.80E-03	**6.78E-01**	1.71E-03
200	20	**6.40E-02**	**3.45E-01**	2.83E-03

among the compared algorithms on instances grouped by problem size and size of agents. We highlight with bold font in Table 6.12 the comparisons that indicate no statistical difference in performance between the algorithms. Obviously, the p values show significant

differences among the algorithmic approaches. However, in some cases, data show no significant difference in some problem instances. In comparing NSGALP and MCRGALP, for instance, over 35% of the problem instances show no significant difference in performance based on the relative hypervolume metric. Likewise, about 78.95% of problem instances show no statistical difference in performance from the epsilon perspective. Specifically, there is no statistical difference in performance between NSGALP and MCRGALP in problem instances of $Size = 100, m = 5$ and $Size = 200, m = 5$ across all four metrics.

Table 6.12 suggests that, despite the exceptional performance of NSGALP, MCRGALP and sequential approaches can also effectively tackle some instances of the linked problem, but that depends on how the two problems are linked. Figure 6.7 gives the overall perspective of the performance of the algorithmic approaches, and the selection of the best approach points toward NSGALP.

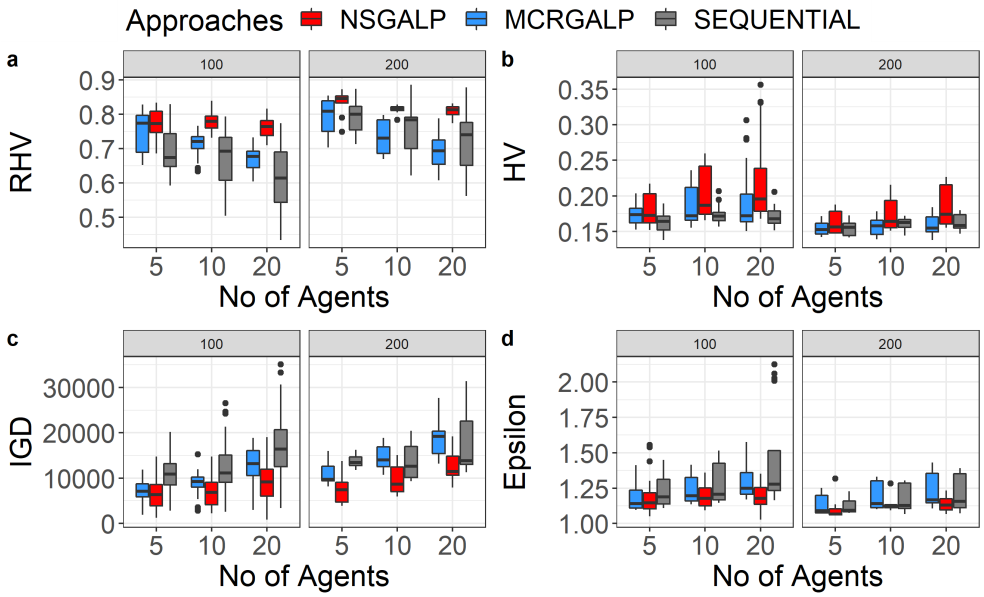


Figure 6.7: Overall performance based on RHV, HV, IGD and Epsilon metrics

Computational Time

We also consider the performance of the algorithmic approaches based on computational time. Figure 6.8 shows four plots of mean computational time against the performance metrics we have used in assessing the performance of the three algorithms. Figure 6.8a shows the mean computing time against the relative hypervolume metric. Figure 6.8b shows the mean computing time against the hypervolume metric, Figure 6.8c IGD and Figure 6.8d shows the mean computing time against the epsilon metric. From the four plots, the grey points represent the mean computing time achieved by the sequential approach, the blue points represent the mean computing time achieved by MCRGALP, and the red points represent the ones achieved by NSGALP. There is no doubt that the sequential approach achieved

less computational time than the other approaches in all combinations of problem instances. It is also interesting to see that, unlike the sequential approach, the computing time achieved by NSGALP and MCRGALP increases with larger problem sizes but is more costly for MCRGALP. This can be seen in the partitions shown by NSGALP and MCRGALP on the four plots. This also suggests that the sequential approach is robust with different problem sizes in terms of computational time. However, the four performance metrics suggest contrary views in most problem instances.

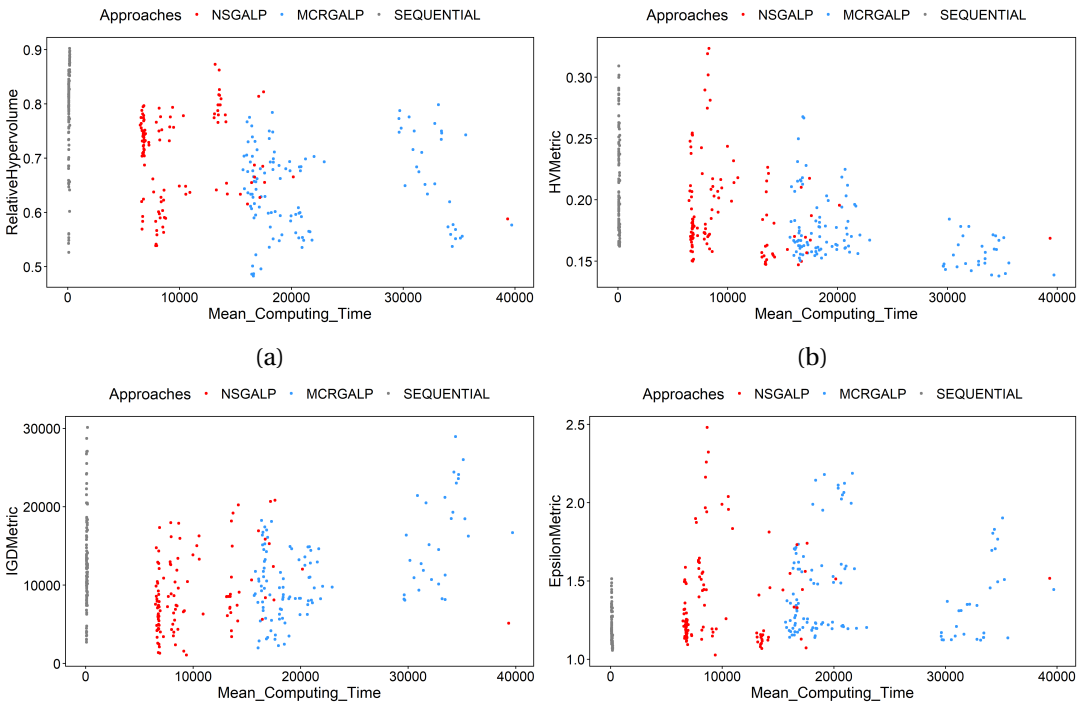


Figure 6.8: Mean Computation Time against Performance Metrics

Correlation Analysis

We further consider algorithm performance in terms of the correlation score obtained by randomly generated solutions. We compare the performance of the competing algorithms on instances that obtained the lowest, median and highest correlation coefficients. See correlation scores in Table 6.13. The essence is to accept the hypothesis that problem instances of JAPTSP which are uncorrelated or are weakly and positively correlated could be tackled by using simple approaches like sequential methods to achieve a joint optimal solution in less computational time. Figure 6.9 shows the empirical attainment function obtained by competing algorithmic approaches on the instances with respective minimum, median and maximum correlation scores. The empirical attainment function (EAF) considers 100 independent experimental runs. Obviously, the correlation scores achieved in all the combined problem instances show no significant linear relationship between JAP and TSP. Figure 6.9a

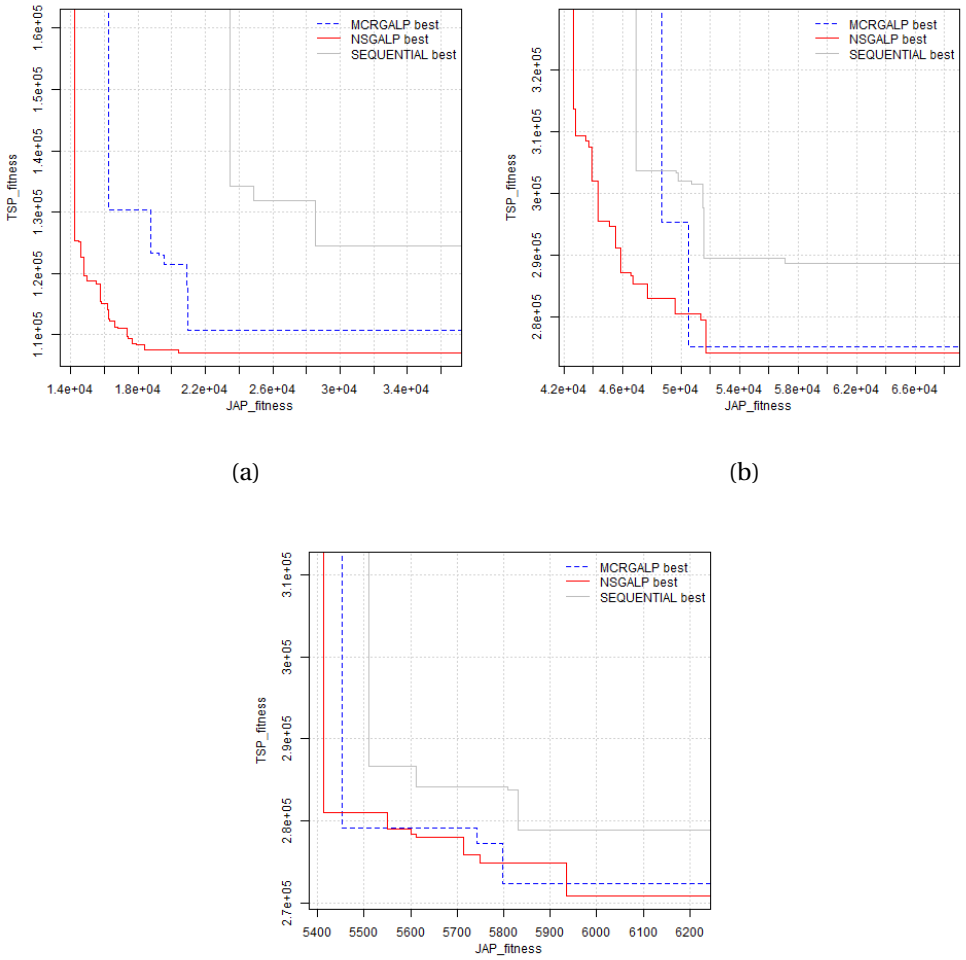
shows the empirical attainment function on the problem instance with the lowest correlation score (-0.0264). With low correlation score of -0.0264 means that, as we try to minimise the JAP cost, there is no significant impact on TSP total travelling distance. There is a low level of trade-off between job cost and travelling distance. Using an NSGALP method in this problem scenario, as seen in Figure 6.9a, would obviously optimise the JAP problem and the TSP problem. Thus, MCRGALP is also efficient in tackling such problem scenarios in terms of coverage. For instance, MCRGALP tries to achieve a solution set that collectively considers the two minimisation problems, although both problems are not fully optimised but are better than sub-optimised points. Figure 6.9b shows how the NSGALP approach produces the best attainment. Thus, solving the JAP problem in such a problem instance scenario would have minimal impact on obtaining an optimal outcome for the TSP problem. This also suggests that an NSGALP approach is more effective, but this takes a considerable amount of time than the sequential method. Figure 6.9c shows the attainment function obtained by the three approaches on a problem instance with a correlation score of 0.0198. The sequential approach could not thrive, which shows that the sequential method performs poorly in terms of coverage, as seen over the three EAF plots in Figure 6.9.

Table 6.13: Problem instances with minimum, median and maximum correlation coefficients

Size	m	Correlation Score
100	20	-0.0264
200	20	-0.0016
200	5	0.0198

Service Chain Perspective

From the perspective of two service companies or business units involved in a service chain, our results can offer guidance on the benefits and costs they are likely to experience based on the approach used to solve the overall problem. For example, the NSGALP is quite interesting due to the large extent of variability in fitness values of the JAP criterion. In Figure 6.7a the NSGALP attempts to solve the first problem in a view to trade-off the other problem. However, there is no obvious trade-off of the second problem as it can be seen that there is a peak at the lower TSP fitness value. In 99% of the problem instances, the MCRGALP tends to hit a sweet spot for both problems, but this does not quantify in terms of the performance metrics used. Therefore, in deciding how to solve the problem with the sequential and NSGALP, it is more apparent that both companies must consider the impact that optimising one problem will have on the other and decide if the resulting costs/benefits are acceptable. In contrast, the MCRGALP maintains a balanced compromise on both problems.



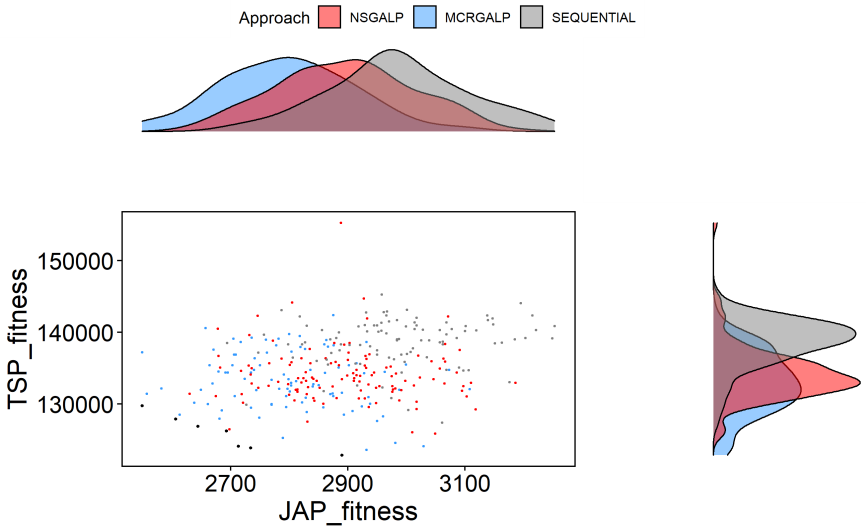
(a) (b)

(c)

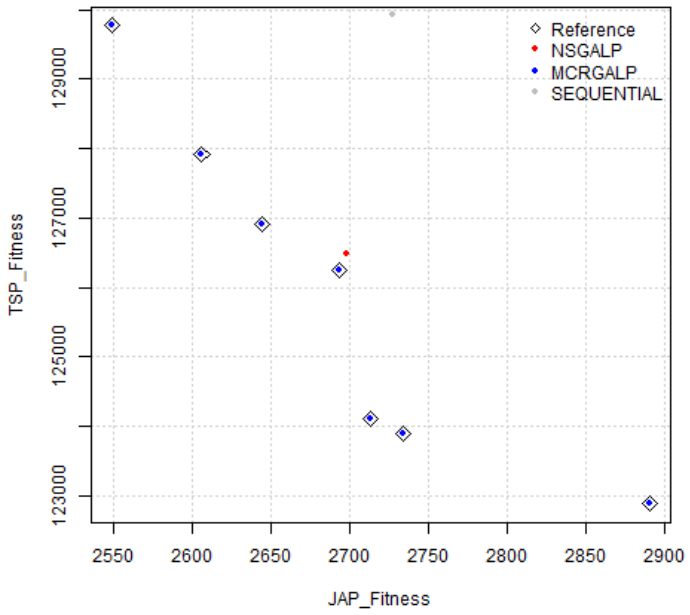
Figure 6.9: Empirical Attainment Function of algorithmic approaches on problem instances with minimum, median and maximum correlation coefficients

6.5 Summary

The contribution in this chapter addresses a linked optimisation of a supply chain by developing and evaluating three algorithmic approaches. Contribution relates to objective (O4) and addresses the research question (RQ4). Similarly, this relates to the supply chain optimisation layer in the methodological framework presented in Section 5.3 of Chapter 5. The chapter considers two case studies of linked optimisation problems (FLPPFSP and JAPTSP) and proposed three algorithmic approaches; NSGALP, MCRGALP, and SEQUENTIAL and adapted them to the linked optimisation framework to solve the two linked problems. FLPPFSP involves the linkages between a facility location problem (FLP) and permutation flow shop scheduling problem (PFSP) of a distributed manufacturing system DMS, and JAPTSP involves the integration between job assignment problem (JAP) and travelling



(a)



(b)

Figure 6.10: Distribution of solutions found by all algorithmic approaches over 100 independent runs on problem size 100 with m=5.

salesman problem (TSP) of a service chain system where service personnel are required to perform tasks at different locations. Experiments were conducted to compare the performance of the three algorithmic approaches and tested on 620 combinations of FLPPFSP and 114 combinations of JAPTSP problem instances respectively. The chapter considers four factors in determining the method's suitability for tackling the two cases of linked problem. These factors include: multi-objective performance metrics selected, mean computation

time required by an algorithm, degree of correlation between the combined problem instance, and qualitative analysis from a service and supply chains perspective.

The empirical results for the FLPPFSP indicate that NSGALP outperforms the other two methods from the perspective of the performance metrics used. However, while testing the significance of the performance, it was observed that 21.7% of problem instances showed no significant difference in performance among the three approaches. It was clear that in terms of mean computational time, the sequential method outperforms the other two methods. In correlation analysis, uncorrelated or weak correlated combined problem instances favour the sequential approach, while the other two methods can solve a negatively correlated problem instance. From the supply chain perspective, reaching a balanced compromise is possible, especially when it involves two or more organisations that have to make conflicting decisions.

The empirical results, in terms of the performance metrics used, indicate that NSGALP outperforms the other two methods. However, statistical test showed that there is no significant difference in performance mostly between NSGALP and MCRGALP. In terms of mean computational time, the sequential method performs best. In the correlation analysis, all the combined problem instances are highly uncorrelated, favouring the NSGALP approach. From the supply chain perspective, MCRGALP seems to maintain balanced multiple decision-making without sacrificing one for the other for each combined problem instance.

7

Conclusion and Future Work

7.1 Research Questions Revisited

(RQ1) *How can we develop a formalism for the linked optimisation problem?*

We develop a general formalism for linked optimisation problems. The formalism allows systems of linked problems to be defined with a precise specification of linkages in terms of solution domain, objective functions and constraints. This was developed in Chapter 3. The formalism is used in exploring linkages between pairs of classical problems and presented different formulations of the linked models in Chapter 5. Results confirmed that different interactions exist with varying levels of relationships which are highly driven by data. As the analysis suggests, the underlying modes of interactions require different algorithmic approaches to reach an overall objective. Therefore, it is important to understand these modes of interaction. Then, from the characteristics that define these interactions, appropriate algorithmic strategies can be implemented in a way that accounts for such characteristics.

(RQ2) *What are the various types of problem linkages that occur in real-world supply chains?*

This thesis identified the problem of linked optimisation in a supply chain. We explored several real-world examples extracted from the literature to validate various modes of linkages in supply chain. This is presented in Chapter 3. To further enhance our understanding of the modes of linkages in supply chain, Chapter 5 presents a set of paired classical problems. The instances of individual problems are identified as mapping to components of supply chain problems. These components allow us to perform an exploratory analysis to ascertain the level of interactions between our selected problem mix. The analysis performed in Chapter 5 suggests that the level of interaction occurs in three modes; the interaction that changes the solution representation of a problem, the interaction that changes the fitness function, and the interaction that changes/modifies the problem constraints. These interactions suggest whether a relationship is linear or non-linear. Specifically, it was also observed that there are variations in the individual linked problem instances, which are highly

driven by the sub-problem instance data. This observation suggests a need for a comprehensive investigation into the linkages of problem instance data.

(RQ3) *How can data relationships be inferred and verified in a supply chain database model?*

This research also addressed discovering data relationships from fragmented supply chain data. We investigated eight relationship discovery algorithms, such as Cosine similarity, Soundex similarity, Name similarity, and Value range similarity, to identify potential links between database tables using different categories of database information. The work relating to RQ3 was presented in Chapter 4.

(RQ4) *What algorithmic methodologies can be designed to tackle complex interactions in supply chain networks?*

In view of tackling complex interactions in supply chain, we considered two different linked problems - FLPPFSP and JAPTSP. We proposed three algorithmic approaches; a sequential approach, a multi-objective approach (NSGALP), and a multi-criteria decision-making (MCRGALP) algorithm. The selection of the three approaches is based on the premise that interacting optimisation problems are connected in diverse ways and may require different and complex algorithmic designs to tackle them. The three algorithmic methods were adopted for linked problems in FLPPFSP and JAPTSP in Chapter 6. The empirical results indicate that NSGALP outperforms the other two methods from the perspective of the performance metrics used in both linked problems. Other factors were also considered in selecting an appropriate method for tackling both FLPPFSP and JAPTSP. These factors include the mean computation time required by an algorithm, the degree of correlation between the combined problem instances, and qualitative analysis from a supply chain perspective. Specifically, in both linked problems, MCRGALP seems to maintain balanced multiple decision-making without sacrificing one for the other. However, the sequential method outperforms the other two methods in terms of computational time. Also, from the correlation analysis, it is seen that uncorrelated or weakly correlated combined problem instances tend to favour the sequential approach.

7.2 Summary of Contributions and Analysis of Limitations

We present a summary of the objectives of this thesis to aid in summarising the contribution of our work.

(O1) To develop a formalism for linked optimisation problem.

(O2) To identify and define a variety of modes of problem linkage that occur in real-world supply chains.

(O3) To investigate a combination of approaches to link fragmented supply chain data.

(O4) To develop and evaluate algorithmic approaches to address linked optimisation problems.

7.2.1 Introduction and formalism of Linked Optimisation Problem

We introduced and developed a general formalism for linked optimisation problems in 3.2.1. The formalism allows the definition of systems of linked problems with a precise specification of linkages in terms of solution domain, objective functions and constraints. Furthermore, research validates this formalism by applying it to several real-world examples extracted from the literature in 3.2.2. This contribution addresses objective (O1), and the work relating to this contribution is in Chapter 3.

7.2.2 Linking fragmented supply chain data

This thesis explores solving fragmented data from linked problems using techniques proposed in the literature. The research investigates how several data relationship discovery algorithms can be combined to improve performance. It investigates eight relationship discovery algorithms - Pseudo-Primary Key Discovery, Cosine similarity, Soundex similarity, Name similarity, Value range similarity, Usage, Semantic Similarity in a Taxonomy, and Content-Based Similarity. These algorithms identify potential links between database tables using different categories of database information. Specifically, the contribution entails adapting a voting system and hierarchical clustering ensemble methods for enhancing discovery performance. The voting scheme uses a given weighting metric to combine the predictions of each algorithm, while the hierarchical clustering groups predictions into clusters based on similarities and then combines a member from each cluster. The result from this investigation suggests that the performance of individual discovery algorithms is limited, indicating the necessity to combine several algorithms to bring together their strengths. Similarly, findings reveal that some specific algorithms are relevant when combined in certain ways and that the choice of algorithms depends mainly on the user's compromise on speed, reliability and sufficiency. This contribution addresses objective (O3), and the work relating to this contribution is in Chapter 4.

7.2.3 Combination of classical problem instances for studying and formulating a variety of Linked Optimisation Problems

The research adopts a method for creating a linked optimisation problem benchmark by linking existing classical benchmark sets. This work uses a mix of classical optimisation problems, typically related to supply chain decision problems, to describe different modes of linkages in linked optimisation problems. The classical problems include; Facility Location Problem (FLP), Knapsack Problem (KP), Job Assignment Problem (JAP), Permutation Flow Shop Scheduling Problem (PFSP), Travelling Salesman Problem (TSP), and Quadratic

Assignment Problem (QAP). This thesis presents different instances of formulations to link the classical optimisation problems to each other. Research ensures a careful selection of a few linkages as these links is exhaustive in real-world applications. These include FLP & JAP, FLP & PFSP, KP & JAP, KP & PFSP, KP & TSP, JAP & TSP, and QAP & TSP. This contribution addresses objective (O2), and the work relating to this contribution is in Chapter 5.

7.2.4 Development of dependency relationships framework between linked problems

This thesis develops a framework for formulating and analysing the dependency relationships between the linked problems. The framework considers the linked problem data linkages, the individual problem domains and the different solution representations for each problem in the linked structure. We relied on benchmark problem instances for the study to test our model. Benchmark problem instances are already well-known problem sets used in developing efficient algorithms. In addition, due to the lack of coherent event logs of supply chain processes, the research relied on human judgement on the perceived dependencies among the components of the linked problem. Therefore, we made an initial process analysis assumption, which already suggests the process sequence of the linked optimisation problem. Contribution addresses objectives (O2) and (O3). Research work relates to the contribution found in Chapters 4 and 5.

7.2.5 Application of three algorithmic methods to tackle Linked Optimisation Problems

The thesis explores methods available to the governing authority for driving the supply chain towards optimising holistic goals. We present three algorithmic approaches to tackling two examples of the linked optimisation problems (FLPPFSP and JAPTSP) formulated in Chapter 5. The approaches include; the Non-dominated Sorting Genetic Algorithm for Linked Problem (NSGALP), Multi-Criteria Ranking Genetic Algorithm for Linked Problem (MCRGALP), and Sequential approach. The exact crossover and mutation operators are used in the three approaches based on a genetic methodological framework. The research adopts a unique procedure for offspring generation for NSGALP and MCRGALP in 6.2.3 and 6.2.4 respectively. NSGALP uses a non-dominated sorting and crowded distance computation embedded in a binary tournament operator that incorporates ranking and diversity. MCRGALP applied a multi-criteria algorithm (TOPSIS) to score each solution pair in a population which is incorporated in a tournament selection operator. The limitation is that the performance metrics used in comparing the performance of the three algorithmic approaches are multi-objective based, which are primarily influenced by the number of non-dominated points produced. Unlike the Pareto set produced by NSGALP, the two other approaches produce a single-point solution pair, and that reduces the chances of them ob-

taining a higher metric score. This contribution addresses objective (O4), and most of the work relating to this contribution is found in Chapter 6.

7.3 Direction for Future Work

7.3.1 Efficient Algorithmic Methods for Complex Supply Chain Structure

This thesis has demonstrated that progress can be achieved by sufficiently adapting algorithms to the different linked structures of a supply chain. In this thesis, our algorithmic methods could scale on linked problems involving two sub-problems but perform differently regarding problem instances. However, our methods have not been tested on a more complex supply chain structure involving three sub-problems. A future consideration on how an algorithmic approach can be efficiently adapted to such a complex structure is worth investigating.

7.3.2 Game Theory

Game theory uses a mathematical concept to study strategic interactions and decision-making among agents [50]. Game theory is conceptualised based on Nash equilibrium or rational behaviour, which results from dynamic processes of adaptation [120]. Game theory involves the rational action of players in a game through repetitive decision-making under the same situation. The idea of a game theory in supply chain optimisation problem is to think of a multi-player cooperative game where the individual objectives, corresponding to each sub-problem to be optimised, is a player in the game and controlled by individual decision-makers. A cooperative game is possible if the players can reach an agreement on strategies [296]. It means that, at the Nash equilibrium action profile, under applicable and straightforward decision-making rules, each player's action is optimal concerning its utility function, given the actions of other players in the game. The game theoretic method was already proposed for multi-objective optimisation problems in [308] where a decision maker controls the action of multi-players. Hence, exploring a more complex network with multiple decision-makers would be interesting.

7.3.3 Process Mining of Complex Supply Chain

There is little research work proposed in the area of supply chain process mining because mining business processes across the supply chain partners (cross-organisation) is a research challenge in the process mining research community [357]. Existing approaches cannot effectively design, operate and evaluate an agile supply chain due to the complex, stochastic, dynamic nature and multi-criteria of logistic processes involved within a global supply chain [191]. A critical layer of the methodological framework proposed in this thesis is the supply chain process model, which was not explored. An investigation of supply

chain process models is anticipated for future research direction since this will account for the agility and holistic process in the supply chain network.

7.3.4 Benchmarking Methodologies

Many examples occur in the literature where particular linked problems are modelled. While these are interesting examples, there is a lack of benchmarks available for the optimisation community to make a systematic study of linked optimisation problems and supply chains. There is a need for methodologies for capturing benchmark problems seen in literature and real-world problems. It is an essential component in creating a platform for investigation. Specifically, investigating a method for linking existing benchmark sets is beneficial. So, for example, given benchmark sets for, say, TSP and KP problems, and a semantic process for how a TSP benchmark solution may give rise to a KP benchmark instance. We can set up a supply chain benchmark where solutions to the linked problems will be pairs (s1, s2) where s1 is a solution to a TSP and s2 is a solution to the KP instantiated from s1.

7.3.5 Linked Problem Dependency Analysis

There is a need for a set of tools for analysing the dependency relationships between the linked problems. Specifically investigating to what extent the choice of solution for a problem P_1 constrains or affects the value available to solvers of the instantiated problem P_2. So that we can determine what set of tools can be selected or implemented for analysing the dependency relationships between the linked problems.

7.3.6 Algorithmic Performance Enhancement

The outstanding performance of the NSGALP and MCRGALP in Chapters 6 and **??** results in sacrificing much computational time in searching for good solution pair. It would be interesting to explore some algorithmic methods properties further to improve efficiency. Thus, developing efficient and different variants of multi-objective algorithms and multi-criteria approaches for linked optimisation and supply chain is another future research direction to be considered.

7.3.7 Appropriate Framework for Holistic Goal

We seek to explore in the future a set of tools to explore methods available to the governing authority for driving the supply chain towards holistic goals. One of the limitations not included in the framework is the idea of a governing authority with holistic goals derived from the solutions adopted by different units across the chain. From a holistic perspective, a network is considered a system that affects its surrounding environment. However, it is not captured in the component problem definitions and may not relate to the component

optimisation goals. Holistic goals may range from corporate-level profitability to environmental goals such as net zero carbon. The consideration of holistic measures will provide an intellectual method for analysing external features underlying the holistic optimisation and by defining the external control in terms of the internal objectives.

7.3.8 Performance Metrics

Concerning future research, it is interesting to consider the use of appropriate performance metrics that measure how algorithms perform towards obtaining results that converge to an equilibrium point (i.e. a balanced joint solution) which is unbiased towards an algorithmic method.

7.3.9 Digital Twins Technology

A digital twin represents a digital form of an asset, process or system [226]. To better have a broader view of a supply chain network, connected digital twins can serve as a potent tool to understand the entire scope of the built environment and the social and environmental layers with which supply chain components interact. These should be considered federated networks of digital twins that span beyond organisational and sectoral silos, thereby connecting processes, information, and organisation to deliver positive outcomes for people, society and nature [226].

Federated networks of digital twins or connected digital twins serve as tools to understand the complexities of interconnected systems and provide better insight to enable better decisions and interventions. It can form a large, evolving ecosystem allowing data sharing to improve understanding of interdependencies between systems to make smarter decisions. Adopting the digital twins framework in supply chain can bring about managing business relationships beyond transactional ones. The adoption will bring about collective knowledge sharing and could establish a broader knowledge-sharing culture, which in turn enhances organisational capabilities [226]. It is worth noting that adopting digital twins to complex supply chain network problems could be an exciting research area to explore.

Bibliography

[1] M. Abdechiri, M. R. Meybodi, and H. Bahrami. Gases brownian motion optimization: an algorithm for optimization (gbmo). *Applied Soft Computing*, 13(5):2932–2946, 2013.

[2] M. Abdel-Basset, L. Abdel-Fatah, and A. Sangaiah. Chapter 10-metaheuristic algorithms: a comprehensive review. computational intelligence for multimedia big data on the cloud with engineering applications, 2018.

[3] R. L. Ackoff. The future of operational research is past. *Journal of the operational research society*, 30(2):93–104, 1979.

[4] S. Ahmad, M. Kamruzzaman, and M. Iqbal. Impacts of optimization in apparel supply chain focusing on ann and genetic algorithm. In *Proceedings of the International Conference on Industrial Engineering and Operations Management*, 2020.

[5] B. Alatas. Chaotic harmony search algorithms. *Applied mathematics and computation*, 216(9):2687–2699, 2010.

[6] J. F. d. F. Almeida, S. V. Conceição, L. R. Pinto, R. S. de Camargo, and G. d. M. Júnior. Flexibility evaluation of multiechelon supply chains. *PloS one*, 13(3):e0194050, 2018.

[7] A. A. Alwan, A. Nordin, M. Alzeber, and A. Z. Abualkishik. A survey of schema matching research using database schemas and instances. *International Journal Of Advanced Computer Science And Applications*, 8(10), 2017.

[8] E. Amrina and A. L. Vilsi. Key performance indicators for sustainable manufacturing evaluation in cement industry. *Procedia Cirp*, 26:19–23, 2015.

[9] E. Angelelli and M. G. Speranza. The periodic vehicle routing problem with intermediate facilities. *European journal of Operational research*, 137(2):233–247, 2002.

[10] R. Ankrah, O. Regnier-Coudert, J. McCall, A. Conway, and A. Hardwick. Performance analysis of ga and pbil variants for real-world location-allocation problems. In *2018 IEEE Congress on Evolutionary Computation (CEC)*, pages 1–8. IEEE, 2018.

[11] S. Arora and M. Puri. A variant of time minimizing assignment problem. *European Journal of Operational Research*, 110(2):314–325, 1998.

[12] S. Ashrafi and A. Dariane. A novel and effective algorithm for numerical optimization: melody search (ms). In *2011 11th international conference on hybrid intelligent systems (HIS)*, pages 109–114. IEEE, 2011.

[13] M. Assaf and M. Ndiaye. Multi travelling salesman problem formulation. In *2017 4th International Conference on Industrial Engineering and Applications (ICIEA)*, pages 292–295. IEEE, 2017.

[14] M. Assi and R. A. Haraty. A survey of the knapsack problem. In *2018 International Arab Conference on Information Technology (ACIT)*, pages 1–6. IEEE, 2018.

[15] E. Atashpaz-Gargari and C. Lucas. Imperialist competitive algorithm: an algorithm for optimization inspired by imperialistic competition. In *2007 IEEE congress on evolutionary computation*, pages 4661–4667. Ieee, 2007.

[16] A. Azzini and P. Ceravolo. Consistent process mining over big data triple stores. In *2013 IEEE International Congress on Big Data*, pages 54–61. IEEE, 2013.

[17] S. Z. A. Bacha, M. W. Belahdji, K. Benatchba, and F. B.-S. Tayeb. A new hyper-heuristic to generate effective instance ga for the permutation flow shop problem. *Procedia Computer Science*, 159:1365–1374, 2019.

[18] J. Bäckstrand. *Levels of interaction in supply chain relations*. PhD thesis, Department of Industrial Engineering and Management, School of Engineering, 2007.

[19] K. Badiru. Knapsack problems; methods, models and applications. 2009.

[20] K. R. Baker and D. Trietsch. *Principles of sequencing and scheduling*. John Wiley & Sons, 2013.

[21] M. Bakker, J. Riezebos, and R. H. Teunter. Review of inventory systems with deterioration since 2001. *European journal of operational research*, 221(2):275–284, 2012.

[22] S. Bandyopadhyay and R. Bhattacharya. Solving multi-objective parallel machine scheduling problem by a modified nsga-ii. *Applied Mathematical Modelling*, 37(10-11):6718–6729, 2013.

[23] E. Bayraktar, S. L. Koh, A. Gunasekaran, K. Sari, and E. Tatoglu. The role of forecasting on bullwhip effect for e-scm applications. *International Journal of Production Economics*, 113(1):193–204, 2008.

[24] J. E. Beasley. Or-library: distributing test problems by electronic mail. *Journal of the operational research society*, 41(11):1069–1072, 1990.

[25] T. Becker and W. Intoyoad. Context aware process mining in logistics. *Procedia Cirp*, 63:557–562, 2017.

[26] Z. Bellahsene, A. Bonifati, and E. Rahm. Schema matching and mapping, section 6. *Data-Centric Systems*, 2011.

[27] U. Benlic, A. E. Brownlee, and E. K. Burke. Heuristic search for the coupled runway sequencing and taxiway routing problem. *Transportation Research Part C: Emerging Technologies*, 71:333–355, 2016.

[28] J. Berlin and A. Motro. Database schema matching using machine learning with feature selection. In *International Conference on Advanced Information Systems Engineering*, pages 452–466. Springer, 2002.

[29] M. L. Bernardi, M. Cimitile, and F. M. Maggi. Discovering cross-organizational business rules from the cloud. In *2014 IEEE Symposium on Computational Intelligence and Data Mining (CIDM)*, pages 389–396. IEEE, 2014.

[30] P. A. Bernstein, J. Madhavan, and E. Rahm. Generic schema matching, ten years later. *Proceedings of the VLDB Endowment*, 4(11):695–701, 2011.

[31] L. C. Bezerra, M. López-Ibánez, and T. Stützle. An empirical assessment of the properties of inverted generational distance on multi-and many-objective optimization. In *International Conference on Evolutionary Multi-Criterion Optimization*, pages 31–45. Springer, 2017.

[32] R. Bhattacharya and S. Bandyopadhyay. Solving conflicting bi-objective facility location problem by nsga ii evolutionary algorithm. *The International Journal of Advanced Manufacturing Technology*, 51(1):397–414, 2010.

[33] M. Bilenko, R. Mooney, W. Cohen, P. Ravikumar, and S. Fienberg. Adaptive name matching in information integration. *IEEE Intelligent Systems*, 18(5):16–23, 2003.

[34] A. Bilke and F. Naumann. Schema matching using duplicates. In *21st International Conference on Data Engineering (ICDE'05)*, pages 69–80. IEEE, 2005.

[35] J. M. Bishop. Stochastic searching networks. In *1989 First IEE international conference on artificial neural networks,(Conf. Publ. No. 313)*, pages 329–331. IET, 1989.

[36] C. Blum and A. Roli. Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM computing surveys (CSUR)*, 35(3):268–308, 2003.

[37] L. Bodin and B. Golden. Classification in vehicle routing and scheduling. *Networks*, 11(2):97–108, 1981.

[38] S. Boettcher and A. G. Percus. Extremal optimization: Methods derived from coevolution. *arXiv preprint math/9904056*, 1999.

[39] P. Bohannon, E. Elnahrawy, W. Fan, and M. Flaster. Putting context into schema matching. In *Proceedings of the 32nd international conference on Very large data bases*, pages 307–318. VLDB Endowment, 2006.

[40] G. Bonilla-Enriquez and S.-O. Caballero-Morales. Search algorithms on logistic and manufacturing problems. 2021.

[41] M. R. Bonyadi and Z. Michalewicz. Evolutionary computation for real-world problems. In *Challenges in Computational Statistics and Data Mining*, pages 1–24. Springer, 2016.

[42] M. R. Bonyadi, Z. Michalewicz, and L. Barone. The travelling thief problem: The first step in the transition from theoretical problems to realistic problems. In *2013 IEEE Congress on Evolutionary Computation*, pages 1037–1044. IEEE, 2013.

[43] M. A. Boschetti, V. Maniezzo, M. Roffilli, and A. Bolufé Röhler. Matheuristics: Optimization, simulation and control. In *International workshop on hybrid metaheuristics*, pages 171–177. Springer, 2009.

[44] D. C. Bose. *Inventory management*. PHI Learning Pvt. Ltd., 2006.

[45] C. C. Bozarth, D. P. Warsing, B. B. Flynn, and E. J. Flynn. The impact of supply chain complexity on manufacturing plant performance. *Journal of Operations Management*, 27(1):78–93, 2009.

[46] J. P. Brans and B. Mareschal. Promethee v: Mcdm problems with segmentation constraints. *INFOR: Information Systems and Operational Research*, 30(2):85–96, 1992.

[47] W. K. M. Brauers and E. K. Zavadskas. Project management by multimoora as an instrument for transition economies. *Technological and economic development of economy*, 16(1):5–24, 2010.

[48] A. Brintrup, Y. Wang, and A. Tiwari. Supply networks as complex systems: a network-science-based characterization. *IEEE Systems Journal*, 11(4):2170–2181, 2015.

[49] J. C. Buijs, B. F. van Dongen, and W. M. van der Aalst. Towards cross-organizational process mining in collections of process models and their executions. In *International Conference on Business Process Management*, pages 2–13. Springer, 2011.

[50] N. Bulling. A survey of multi-agent decision making. *KI-Künstliche Intelligenz*, 28(3):147–158, 2014.

[51] R. E. Burkard, S. E. Karisch, and F. Rendl. Qaplib–a quadratic assignment problem library. *Journal of Global optimization*, 10(4):391–403, 1997.

[52] R. E. Burkard and F. Rendl. Lexicographic bottleneck problems. *Operations Research Letters*, 10(5):303–308, 1991.

[53] P. J. Burke. An identity model for network exchange. *American Sociological Review*, pages 134–150, 1997.

[54] S. F. M. Burnet et al. *The clonal selection theory of acquired immunity*, volume 3. Vanderbilt University Press Nashville, 1959.

[55] R. T. Bye, M. Gribbestad, R. Chandra, and O. L. Osen. A comparison of ga crossover and mutation methods for the traveling salesman problem. In *Innovations in Computational Intelligence and Computer Vision: Proceedings of ICICV 2020*, pages 529–542. Springer, 2021.

[56] F. Camci. The travelling maintainer problem: integration of condition-based maintenance with the travelling salesman problem. *Journal of the Operational Research Society*, 65(9):1423–1436, 2014.

[57] A. Campbell, L. Clarke, A. Kleywegt, and M. Savelsbergh. The inventory routing problem. In *Fleet management and logistics*, pages 95–113. Springer, 1998.

[58] Ü. Can and B. Alataş. Physics based metaheuristic algorithms for global optimization. 2015.

[59] M. Caramia and P. Dell'Olmo. Multi-objective optimization. In *Multi-objective management in freight logistics*, pages 21–51. Springer, 2020.

[60] G. Caron, P. Hansen, and B. Jaumard. The assignment problem with seniority and job priority constraints. *Operations Research*, 47(3):449–453, 1999.

[61] S. Castano and V. De Antonellis. Global viewing of heterogeneous data sources. *IEEE Transactions on Knowledge and Data Engineering*, 13(2):277–297, 2001.

[62] A. Castillo-Salazar, D. Landa-Silva, and R. Qu. A survey of workforce scheduling and routing. 2012.

[63] J. A. Castillo-Salazar, D. Landa-Silva, and R. Qu. A greedy heuristic for workforce scheduling and routing with time-dependent activities constraints. 2015.

[64] D. G. Cattrysse and L. N. Van Wassenhove. A survey of algorithms for the generalized assignment problem. *European journal of operational research*, 60(3):260–272, 1992.

[65] B. Ceballos, M. T. Lamata, and D. A. Pelta. A comparative analysis of multi-criteria decision-making methods. *Progress in Artificial Intelligence*, 5(4):315–322, 2016.

[66] H. Ch and K. Yoon. Multiple attribute decision making. methods and applications, 1981.

[67] F. T. Chan, S. H. Chung, and S. Wadhwa. A hybrid genetic algorithm for production and distribution. *Omega*, 33(4):345–355, 2005.

[68] P. Chandra and M. L. Fisher. Coordination of production and distribution planning. *European Journal of Operational Research*, 72(3):503–517, 1994.

[69] G. J. Chang and P.-H. Ho. The β-assignment problems. *European Journal of Operational Research*, 104(3):593–600, 1998.

[70] V. Chankong and Y. Y. Haimes. Optimization-based methods for multiobjective decision-making-an overview. *Large Scale Systems In Information And Decision Technologies*, 5(1):1–33, 1983.

[71] M. Chavent, V. Kuentz, B. Liquet, and L. Saracco. Clustofvar: an r package for the clustering of variables. *arXiv preprint arXiv:1112.0295*, 2011.

[72] C.-L. Chen and W.-C. Lee. Multi-objective optimization of multi-echelon supply chain networks with uncertain product demands and prices. *Computers & Chemical Engineering*, 28(6-7):1131–1144, 2004.

[73] L. Chen, A. Langevin, and Z. Lu. Integrated scheduling of crane handling and truck transportation in a maritime container terminal. *European Journal of Operational Research*, 225(1):142–152, 2013.

[74] T. Chen. A novel bionic intelligent optimization algorithm: Artificial tribe algorithm and its performance analysis. In *2010 international conference on measuring technology and mechatronics automation*, volume 1, pages 222–225. IEEE, 2010.

[75] T.-L. Chen, C.-Y. Cheng, Y.-Y. Chen, and L.-K. Chan. An efficient hybrid algorithm for integrated order batching, sequencing and routing problem. *International Journal of Production Economics*, 159:158–167, 2015.

[76] Z. Chen, V. Narasayya, and S. Chaudhuri. Fast foreign-key detection in microsoft sql server powerpivot for excel. *Proceedings of the VLDB Endowment*, 7(13):1417–1428, 2014.

[77] K. Chin and H. Lee. Analysis of the vulnerabilities of the supply chain network of a manufacturing company using the network-science approach: S-electronics case. In *2018 Portland International Conference on Management of Engineering and Technology (PICMET)*, pages 1–6. IEEE, 2018.

[78] T. A. Chin, H. H. Tat, and Z. Sulaiman. Green supply chain management, environmental collaboration and sustainability performance. *Procedia Cirp*, 26:695–699, 2015.

[79] T. Y. Choi, K. J. Dooley, and M. Rungtusanatham. Supply networks and complex adaptive systems: control versus emergence. *Journal of operations management*, 19(3):351–366, 2001.

[80] J. Chow. *Informed Urban transport systems: Classic and emerging mobility methods toward smart cities*. Elsevier, 2018.

[81] P. Christen. *Data matching: concepts and techniques for record linkage, entity resolution, and duplicate detection*. Springer Science & Business Media, 2012.

[82] M. Christopher and M. Holweg. "supply chain 2.0": Managing supply chains in the era of turbulence. *International journal of physical distribution & logistics management*, 41(1):63–82, 2011.

[83] C. E. H. Chua, R. H. Chiang, and E.-P. Lim. Instance-based attribute identification in database integration. *The VLDB Journal—The International Journal on Very Large Data Bases*, 12(3):228–243, 2003.

[84] J. Claes and G. Poels. Process mining and the prom framework: an exploratory survey. In *International Conference on Business Process Management*, pages 187–198. Springer, 2012.

[85] J. Claes and G. Poels. Merging event logs for process mining: A rule based merging method and rule suggestion algorithm. *Expert Systems with Applications*, 41(16):7291–7306, 2014.

[86] G. Clarke and J. W. Wright. Scheduling of vehicles from a central depot to a number of delivery points. *Operations research*, 12(4):568–581, 1964.

[87] W. W. Cohen, H. Kautz, and D. McAllester. Hardening soft information sources. In *KDD*, pages 255–259, 2000.

[88] J. L. Cohon. Multicriteria programming: Brief review and application. *Design optimization*, pages 163–191, 1985.

[89] A. Colorni, M. Dorigo, V. Maniezzo, et al. Distributed optimization by ant colonies. In *Proceedings of the first European conference on artificial life*, volume 142, pages 134–142. Paris, France, 1991.

[90] M. Comuzzi, I. Vanderfeesten, and T. Wang. Optimized cross-organizational business process monitoring: Design and enactment. *Information Sciences*, 244:107–118, 2013.

[91] G. Conti and A. Dow. The impacts of covid-19 on health visiting services in england: Foi evidence for the first wave. 2020.

[92] K. S. Cook and J. M. Whitmeyer. Two approaches to social structure: Exchange theory and network analysis. *Annual review of Sociology*, 18(1):109–127, 1992.

[93] L. Cooper. Location-allocation problems. *Operations research*, 11(3):331–343, 1963.

[94] J.-F. Cordeau, G. Laporte, J.-Y. Potvin, and M. W. Savelsbergh. Transportation on demand. *Handbooks in operations research and management science*, 14:429–466, 2007.

[95] H. Corley. A new scalar equivalence for pareto optimization. *IEEE Transactions on Automatic Control*, 25(4):829–830, 1980.

[96] A. Crarnes, W. Cooper, J. Harrald, K. Karwan, and W. Wallace. A goal interval programming model for resource allocation in a marine environmental protection program. *Journal of Environmental Economics and Management*, 3(4):347–362, 1976.

[97] G. B. Dantzig and J. H. Ramser. The truck dispatching problem. *Management science*, 6(1):80–91, 1959.

[98] P. Daoutidis, W. Tang, and A. Allman. Decomposition of control and optimization problems by network structure: Concepts, methods, and inspirations from biology. *AIChE Journal*, 65(10):e16708, 2019.

[99] D. G. de Barros Franco, M. T. A. Steiner, and F. M. Assef. Optimization in waste landfilling partitioning in paraná state, brazil. *Journal of Cleaner Production*, 283:125353, 2021.

[100] M. G. De Carvalho, A. H. Laender, M. A. GonçAlves, and A. S. Da Silva. An evolutionary approach to complex schema matching. *Information Systems*, 38(3):302–316, 2013.

[101] L. N. de Castro and F. J. Von Zuben. ainet: an artificial immune network for data analysis. In *Data mining: a heuristic approach*, pages 231–260. IGI Global, 2002.

[102] K. Deb. Multi-objective optimization. In *Search methodologies*, pages 403–449. Springer, 2014.

[103] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE transactions on evolutionary computation*, 6(2):182–197, 2002.

[104] M. Dees, M. de Leoni, and F. Mannhardt. Enhancing process models to improve business performance: A methodology and case studies. In *On the Move to Meaningful Internet Systems. OTM 2017 Conferences: Confederated International Conferences: CoopIS, C&TC, and ODBASE 2017, Rhodes, Greece, October 23-27, 2017, Proceedings, Part I*, pages 232–251. Springer, 2017.

[105] J. Del Ser, E. Osaba, D. Molina, X.-S. Yang, S. Salcedo-Sanz, D. Camacho, S. Das, P. N. Suganthan, C. A. C. Coello, and F. Herrera. Bio-inspired computation: Where we stand and what's next. *Swarm and Evolutionary Computation*, 48:220–250, 2019.

[106] M. Dell'Amico and S. Martello. The k-cardinality assignment problem. *Discrete Applied Mathematics*, 76(1-3):103–121, 1997.

[107] J. Deng and L. Wang. A competitive memetic algorithm for multi-objective distributed permutation flow shop scheduling problem. *Swarm and evolutionary computation*, 32:121–131, 2017.

[108] G. Ding, H. Dong, and G. Wang. Appearance-order-based schema matching. In *International Conference on Database Systems for Advanced Applications*, pages 79–94. Springer, 2012.

[109] G. Ding, T. Sun, and Y. Xu. Multi-schema matching based on clustering techniques. In *2013 10th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD)*, pages 778–782. IEEE, 2013.

[110] H. Ding, Q. Zhao, Z. An, and O. Tang. Collaborative mechanism of a sustainable supply chain with environmental constraints and carbon caps. *International Journal of Production Economics*, 181:191–207, 2016.

[111] H.-H. Do. Schema matching and mapping-based data integration. 2006.

[112] H.-H. Do and E. Rahm. Coma: a system for flexible combination of schema matching approaches. In *Proceedings of the 28th international conference on Very Large Data Bases*, pages 610–621. VLDB Endowment, 2002.

[113] H.-H. Do and E. Rahm. Matching large schemas: Approaches and evaluation. *Information Systems*, 32(6):857–885, 2007.

[114] A. Doan, P. Domingos, and A. Y. Halevy. Reconciling schemas of disparate data sources: A machine-learning approach. In *ACM Sigmod Record*, volume 30, pages 509–520. ACM, 2001.

[115] A. Doan, P. M. Domingos, and A. Y. Levy. Learning source description for data integration. In *WebDB (Informal Proceedings)*, pages 81–86, 2000.

[116] P. Domingos, D. Lowd, S. Kok, A. Nath, H. Poon, M. Richardson, and P. Singla. Markov logic: a language and algorithms for link mining. In *Link Mining: Models, Algorithms, and Applications*, pages 135–161. Springer, 2010.

[117] M. Dorigo, V. Maniezzo, and A. Colorni. The ant system: An autocatalytic optimizing process. 1991.

[118] F. Duchateau, Z. Bellahsene, M. Roantree, and M. Roche. An indexing structure for automatic schema matching. In *2007 IEEE 23rd International Conference on Data Engineering Workshop*, pages 485–491. IEEE, 2007.

[119] J. J. Durillo and A. J. Nebro. jmetal: A java framework for multi-objective optimization. *Advances in Engineering Software*, 42(10):760–771, 2011.

[120] C. Eksin, B. Swenson, S. Kar, and A. Ribeiro. Game theoretic learning. In *Cooperative and Graph Signal Processing*, pages 209–235. Elsevier, 2018.

[121] B. Eksioglu, A. V. Vural, and A. Reisman. The vehicle routing problem: A taxonomic review. *Computers & Industrial Engineering*, 57(4):1472–1483, 2009.

[122] H. Elmeleegy, M. Ouzzani, and A. Elmagarmid. Usage-based schema matching. In *2008 IEEE 24th International Conference on Data Engineering*, pages 20–29. IEEE, 2008.

[123] R. Engel, W. Krathu, M. Zapletal, C. Pichler, R. J. C. Bose, W. van der Aalst, H. Werthner, and C. Huemer. Analyzing inter-organizational business processes. *Information Systems and e-Business Management*, 14(3):577–612, 2016.

[124] R. Engel, W. M. van der Aalst, M. Zapletal, C. Pichler, and H. Werthner. Mining inter-organizational business process models from edi messages: A case study from the automotive sector. In *International Conference on Advanced Information Systems Engineering*, pages 222–237. Springer, 2012.

[125] K. Ertogral, S. D. Wu, and L. I. Burke. Coordination production and transportation scheduling in the supply chain. *Dept. IMSE, Lehigh Univ., Bethlehem, PA, Tech. Rep*, 1998.

[126] E. Fadakar and M. Ebrahimi. A new metaheuristic football game inspired algorithm. In *2016 1st conference on swarm intelligence and evolutionary computation (CSIEC)*, pages 6–11. IEEE, 2016.

[127] M. A. Falcone, H. S. Lopes, and L. dos Santos Coelho. Supply chain optimisation using evolutionary algorithms. *International Journal of Computer Applications in Technology*, 31(3-4):158–167, 2008.

[128] R. Z. Farahani, N. Asgari, N. Heidari, M. Hosseininia, and M. Goh. Covering problems in facility location: A review. *Computers & Industrial Engineering*, 62(1):368–407, 2012.

[129] J. D. Farmer, N. H. Packard, and A. S. Perelson. The immune system, adaptation, and machine learning. *Physica D: Nonlinear Phenomena*, 22(1-3):187–204, 1986.

[130] S. Fatemi Ghomi, B. Karimi, J. Behnamian, and J. Firoozbakht. A multi-objective particle swarm optimization based on pareto archive for integrated production and distribution planning in a green supply chain. *Applied Artificial Intelligence*, 35(2):133–153, 2021.

[131] I. P. Fellegi and A. B. Sunter. A theory for record linkage. *Journal of the American Statistical Association*, 64(328):1183–1210, 1969.

[132] J. Feng, X. Hong, and Y. Qu. An instance-based schema matching method with attributes ranking and classification. In *2009 Sixth International Conference on Fuzzy Systems and Knowledge Discovery*, volume 5, pages 522–526. IEEE, 2009.

[133] T. A. Feo and M. G. Resende. Greedy randomized adaptive search procedures. *Journal of global optimization*, 6(2):109–133, 1995.

[134] V. Fernandez-Viagas, P. Perez-Gonzalez, and J. M. Framinan. The distributed permutation flow shop to minimise the total flowtime. *Computers & Industrial Engineering*, 118:464–477, 2018.

[135] P. C. Fishburn. Exceptional paper—lexicographic orders, utilities and decision rules: A survey. *Management science*, 20(11):1442–1471, 1974.

[136] L. J. Fogel, A. J. Owens, and M. J. Walsh. Evolution of finite automata for prediction. Technical report, GENERAL DYNAMICS SAN DIEGO CA CONVAIR DIV, 1966.

[137] C. M. Fonseca, A. P. Guerreiro, M. López-Ibánez, and L. Paquete. On the computation of the empirical attainment function. In *International Conference on Evolutionary Multi-criterion Optimization*, pages 106–120. Springer, 2011.

[138] R. A. Formato. Central force optimization. *Prog Electromagn Res*, 77(1):425–491, 2007.

[139] L. Foulds and J. M. Wilson. On an assignment problem with side constraints. *Computers & industrial engineering*, 37(4):847–858, 1999.

[140] L. C. Freeman. A set of measures of centrality based on betweenness. *Sociometry*, pages 35–41, 1977.

[141] Y. Fu, G. Tian, A. M. Fathollahi-Fard, A. Ahmadi, and C. Zhang. Stochastic multi-objective modelling and optimization of an energy-conscious distributed permutation flow shop scheduling problem with the total tardiness constraint. *Journal of cleaner production*, 226:515–525, 2019.

[142] A. H. Gandomi, X.-S. Yang, and A. H. Alavi. Mixed variable structural optimization using firefly algorithm. *Computers & Structures*, 89(23-24):2325–2336, 2011.

[143] M. Ganesh, J. Srivastava, and T. Richardson. Mining entity-identification rules for database integration. In *KDD*, volume 96, pages 2–4, 1996.

[144] J. Gao and R. Chen. A hybrid genetic algorithm for the distributed permutation flowshop scheduling problem. *International Journal of Computational Intelligence Systems*, 4(4):497–508, 2011.

[145] J. Gao, R. Chen, and W. Deng. An efficient tabu search algorithm for the distributed permutation flowshop scheduling problem. *International Journal of Production Research*, 51(3):641–651, 2013.

[146] S. Gass and T. Saaty. The computational algorithm for the parametric objective function. *Naval research logistics quarterly*, 2(1-2):39–45, 1955.

[147] B. Gavish and S. C. Graves. The travelling salesman problem and related problems. 1978.

[148] Z. W. Geem, J. H. Kim, and G. V. Loganathan. A new heuristic optimization algorithm: harmony search. *simulation*, 76(2):60–68, 2001.

[149] S. Geetha and K. Nair. A variation of the assignment problem. *European Journal of Operational Research*, 68(3):422–426, 1993.

[150] T. Geetha and K. Muthukumaran. An observational analysis of genetic operators. *International Journal of Computer Applications*, 63(18):24–34, 2013.

[151] R. Gerhard. The traveling salesman: computational solutions for tsp applications. *Lecture Notes in Computer Science*, 840:1–223, 1994.

[152] K. Gerke, A. Claus, and J. Mendling. Process mining of rfid-based supply chains. In *2009 IEEE Conference on Commerce and Enterprise Computing*, pages 285–292. IEEE, 2009.

[153] F. Glover. Heuristics for integer programming using surrogate constraints. *Decision sciences*, 8(1):156–166, 1977.

[154] F. Glover. Future paths for integer programming and links to artificial intelligence. *Computers & operations research*, 13(5):533–549, 1986.

[155] F. Glover and C. McMillan. The general employee scheduling problem. an integration of ms and ai. *Computers & operations research*, 13(5):563–573, 1986.

[156] O. Gross. The bottleneck assignment problem. Technical report, RAND CORP SANTA MONICA CALIF, 1959.

[157] G. Grygiel. Algebraic k-assignment problems. *Control and Cybernetics*, 10(3-4):155–165, 1981.

[158] A. Gupta, Y.-S. Ong, and L. Feng. Multifactorial evolution: toward evolutionary multitasking. *IEEE Transactions on Evolutionary Computation*, 20(3):343–357, 2015.

[159] S. Gupta and A. Punnen. Minimum deviation problems. *Operations research letters*, 7(4):201–204, 1988.

[160] V. Gutierrez and C. J. Vidal. Inventory management models in supply chains: A literature review. *Revista Facultad de Ingeniería Universidad de Antioquia*, (43):134–149, 2008.

[161] D. H. Hai. Schema matching and mapping-based data integration. *University of Leipzig*, 2005.

[162] Y. Haimes. On a bicriterion formulation of the problems of integrated system identification and system optimization. *IEEE transactions on systems, man, and cybernetics*, 1(3):296–297, 1971.

[163] A. Halevy, M. Franklin, and D. Maier. Principles of dataspace systems. In *Proceedings of the twenty-fifth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 1–9. ACM, 2006.

[164] A. Halevy, A. Rajaraman, and J. Ordille. Data integration: the teenage years. In *Proceedings of the 32nd international conference on Very large data bases*, pages 9–16. VLDB Endowment, 2006.

[165] C. M. Harland. Supply chain management: relationships, chains and networks. *British Journal of management*, 7:S63–S80, 1996.

[166] I. Harris, C. Mumford, and M. Naim. The multi-objective uncapacitated facility location problem for green logistics. In *2009 IEEE Congress on Evolutionary Computation*, pages 2732–2739. IEEE, 2009.

[167] Z. He, Z. Guo, and J. Wang. Integrated scheduling of production and distribution operations in a global mto supply chain. *Enterprise Information Systems*, 13(4):490–514, 2019.

[168] J. Heizer and B. Render. Operations management, 2006.

[169] A. Hemmati, M. Stålhane, L. M. Hvattum, and H. Andersson. An effective heuristic for solving a combined cargo and inventory routing problem in tramp shipping. *Computers & Operations Research*, 64:274–282, 2015.

[170] M. A. Hernández and S. J. Stolfo. Real-world data is dirty: Data cleansing and the merge/purge problem. *Data mining and knowledge discovery*, 2(1):9–37, 1998.

[171] M. Herschel, R. Diestelkämper, and H. Ben Lahmar. A survey on provenance: What for? what form? what from? *The VLDB Journal*, 26(6):881–906, 2017.

[172] F. S. Hillier and M. M. Connors. Quadratic assignment problem algorithms and the location of indivisible facilities. *Management Science*, 13(1):42–57, 1966.

[173] W. Ho, G. T. Ho, P. Ji, and H. C. Lau. A hybrid genetic algorithm for the multi-depot vehicle routing problem. *Engineering applications of artificial intelligence*, 21(4):548–557, 2008.

[174] J. H. Holland. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press, 1992.

[175] K. Holmberg, M. Rönnqvist, and D. Yuan. An exact algorithm for the capacitated facility location problems with single sourcing. *European Journal of Operational Research*, 113(3):544–559, 1999.

[176] B. Hompes, J. C. Buijs, W. M. van der Aalst, P. M. Dixit, and J. Buurman. Detecting changes in process behavior using comparative case clustering. In *Data-Driven Process Discovery and Analysis: 5th IFIP WG 2.6 International Symposium, SIMPDA 2015, Vienna, Austria, December 9-11, 2015, Revised Selected Papers 5*, pages 54–75. Springer, 2017.

[177] H. S. Hosseini. Problem solving by intelligent water drops. In *2007 IEEE congress on evolutionary computation*, pages 3226–3231. IEEE, 2007.

[178] X. Hu, C. Liang, D. Chang, and Y. Zhang. Container storage space assignment problem in two terminals with the consideration of yard sharing. *Advanced Engineering Informatics*, 47:101224, 2021.

[179] J. Huang, Q. Pan, and Q. Chen. A hybrid genetic algorithm for the distributed permutation flowshop scheduling problem with sequence-dependent setup times. In *IOP Conference Series: Materials Science and Engineering*, volume 646, page 012037. IOP Publishing, 2019.

[180] C.-L. Hwang and A. S. M. Masud. Methods for multiple objective decision making. In *Multiple objective decision making—methods and applications*, pages 21–283. Springer, 1979.

[181] Z. Ibrahim, N. H. A. Aziz, N. A. A. Aziz, S. Razali, and M. S. Mohamad. Simulated kalman filter: a novel estimation-based metaheuristic optimization algorithm. *Advanced Science Letters*, 22(10):2941–2946, 2016.

[182] M. Ibrahimov. *Evolutionary algorithms for supply chain optimisation*. PhD thesis, 2012.

[183] M. Ibrahimov, A. Mohais, and Z. Michalewicz. Global optimization in supply chain operations. In *Natural Intelligence for Scheduling, Planning and Packing Problems*, pages 1–28. Springer, 2009.

[184] M. Ibrahimov, A. Mohais, S. Schellenberg, and Z. Michalewicz. Evolutionary approaches for supply chain optimisation: part i: single and two-component supply chains. *International Journal of Intelligent Computing and Cybernetics*, 5(4):444–472, 2012a.

[185] P. Ignaciuk and Ł. Wieczorek. Continuous genetic algorithms in the optimization of logistic networks: Applicability assessment and tuning. *Applied Sciences*, 10(21):7851, 2020.

[186] M. Iori and S. Martello. Routing problems with loading constraints. *Top*, 18(1):4–27, 2010.

[187] R. Irizarry. Fuzzy classification with an artificial chemical process. *Chemical engineering science*, 60(2):399–412, 2005.

[188] H. Irshad, B. Shafiq, J. Vaidya, M. A. Bashir, S. Shamail, and N. Adam. Preserving privacy in collaborative business process composition. In *2015 12th International Joint Conference on e-Business and Telecommunications (ICETE)*, volume 4, pages 112–123. IEEE, 2015.

[189] F. István and H. Tamás. Network science in logistics: A new way to flexible adaptation. In *Flexibility in resource management*, pages 57–69. Springer, 2018.

[190] S. Ito, D. Vymětal, R. Šperka, and M. Halaška. Process mining of a multi-agent business simulator. *Computational and Mathematical Organization Theory*, 24(4):500–531, 2018.

[191] V. Jain, L. Benyoucef, and S. Deshmukh. Strategic supplier selection: some emerging issues and challenges. *International Journal of Logistics Systems and Management*, 5(1-2):61–88, 2008.

[192] G. Janssenswillen, B. Depaire, and S. Verboven. Detecting train reroutings with process mining: A belgian application. *EURO Journal on Transportation and Logistics*, 7(1):1–24, 2018.

[193] M. A. Jaro. Advances in record-linkage methodology as applied to matching the 1985 census of tampa, florida. *Journal of the American Statistical Association*, 84(406):414–420, 1989.

[194] A. Jayant and J. Sharma. A comprehensive literature review of mcdm techniques electre, promethee, vikor and topsis applications in business competitive environment. *International Journal of Current Research*, 10(2):65461–65477, 2018.

[195] B. Ji, H. Huang, and S. Y. Samson. An enhanced nsga-ii for solving berth allocation and quay crane assignment problem with stochastic arrival times. *IEEE Transactions on Intelligent Transportation Systems*, 2022.

[196] F. Ji, H. Xiaoguang, and Q. Yuanbo. An instance-based schema matching method with attributes ranking and classification. In *Proceedings of the 6th International Conference on Fuzzy Systems and Knowledge Discovery FSKD'09*, pages 14–16, 2009.

[197] P. Ji and W. Ho. The traveling salesman and the quadratic assignment problems: Integration, modeling and genetic algorithm. In *Proceedings of International symposium on OR and its applications*. Citeseer, 2005.

[198] L. Jiang and F. Naumann. Holistic primary key and foreign key detection. *Journal of Intelligent Information Systems*, pages 1–23, 2019.

[199] B. Jokonowo, J. Claes, R. Sarno, and S. Rochimah. Process mining in supply chains: A systematic literature review. *International Journal of Electrical and Computer Engineering*, 8(6), 2018.

[200] A. Jozwiak, M. Milkovics, and Z. Lakner. A network-science support system for food chain safety: A case from hungarian cattle production. *International Food and Agribusiness Management Review*, 19(1030-2016-83145):1–26, 2016.

[201] M. Jünger, G. Reinelt, and G. Rinaldi. The traveling salesman problem. *Handbooks in operations research and management science*, 7:225–330, 1995.

[202] J. Kang and J. F. Naughton. On schema matching with opaque column names and data values. In *Proceedings of the 2003 ACM SIGMOD international conference on Management of data*, pages 205–216. ACM, 2003.

[203] D. Karaboga et al. An idea based on honey bee swarm for numerical optimization. Technical report, Technical report-tr06, Erciyes university, engineering faculty, computer . . . , 2005.

[204] A. H. Kashan. League championship algorithm (lca): An algorithm for global optimization inspired by sport championships. *Applied Soft Computing*, 16:171–200, 2014.

[205] A. Kaveh and S. Talatahari. A novel heuristic optimization method: charged system search. *Acta mechanica*, 213(3):267–289, 2010.

[206] A. Kaveh and A. Zolghadr. A novel meta-heuristic algorithm: tug of war optimization. *Iran University of Science & Technology*, 6(4):469–492, 2016.

[207] R. Keeney and H. Raiffa. Decisions with multiple objectives: Preferences and value tradeoffs john wiley & sons new york. 1976.

[208] H. Kellerer, U. Pferschy, and D. Pisinger. Multiple knapsack problems. In *Knapsack Problems*, pages 285–316. Springer, 2004.

[209] J. Kennedy and R. Eberhart. Particle swarm optimization. In *Proceedings of ICNN'95-international conference on neural networks*, volume 4, pages 1942–1948. IEEE, 1995.

[210] P. Ketikidis, S. Koh, N. Dimitriadis, A. Gunasekaran, and M. Kehajova. The use of information systems for logistics and supply chain management in south east europe: Current status and future direction. *Omega*, 36(4):592–599, 2008.

[211] A. Khan, A. Lodhi, V. Köppen, G. Kassem, and G. Saake. Applying process mining in soa environments. In *Service-Oriented Computing. ICSOC/ServiceWave 2009 Workshops*, pages 293–302. Springer, 2009.

[212] S. A. Khan, A. Chaabane, and F. T. Dweiri. Multi-criteria decision-making methods application in supply chain management: A systematic literature. *Multi-criteria methods and techniques applied to supply chain management*, 1:10–5772, 2018.

[213] K. Kim and C. A. Ellis. σ-algorithm: Structured workflow process mining through amalgamating temporal workcases. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 119–130. Springer, 2007.

[214] S. Kirkpatrick, C. D. Gelatt Jr, and M. P. Vecchi. Optimization by simulated annealing. *science*, 220(4598):671–680, 1983.

[215] D. Knoll, G. Reinhart, and M. Prüglmeier. Enabling value stream mapping for internal logistics using multidimensional process mining. *Expert Systems with Applications*, 124:130–142, 2019.

[216] A. Konak, S. Kulturel-Konak, and L. Snyder. A multi-objective approach to the competitive facility location problem. *Procedia Computer Science*, 108:1434–1442, 2017.

[217] L. Kong, J. Wang, and P. Zhao. Solving the dynamic weapon target assignment problem by an improved multiobjective particle swarm optimization algorithm. *Applied Sciences*, 11(19):9254, 2021.

[218] V. M. Korac, J. Kratica, and A. Savić. An improved genetic algorithm for the multi level uncapacitated facility location problem. *International Journal of Computers Communications & Control*, 8(6):845–853, 2013.

[219] V. Kotu and B. Deshpande. *Data Science: Concepts and Practice*. Morgan Kaufmann, 2018.

[220] P. Kouvelis and G. Yu. *Robust discrete optimization and its applications*, volume 14. Springer Science & Business Media, 2013.

[221] J. R. Koza et al. *Genetic programming II*, volume 17. MIT press Cambridge, 1994.

[222] J. Kratica, D. Tošic, V. Filipović, and I. Ljubić. Solving the simple plant location problem by genetic algorithm. *RAIRO-Operations Research*, 35(1):127–142, 2001.

[223] K. N. Krishnanand and D. Ghose. Detection of multiple source locations using a glowworm metaphor with applications to collective robotics. In *Proceedings 2005 IEEE Swarm Intelligence Symposium, 2005. SIS 2005.*, pages 84–91. IEEE, 2005.

[224] H. W. Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.

[225] A. Y. Lam and V. O. Li. Chemical-reaction-inspired metaheuristic for optimization. *IEEE transactions on evolutionary computation*, 14(3):381–399, 2009.

[226] K. Lamb. Gemini papers: How to enable an ecosystem of connected digital twins. 2022.

[227] H. C. Lau, G. T. Ho, Y. Zhao, and N. Chung. Development of a process mining system for supporting knowledge discovery in a supply chain network. *International Journal of Production Economics*, 122(1):176–187, 2009.

[228] H. C. Lau and Y. Song. Combining two heuristics to solve a supply chain optimization problem. 2002.

[229] T. D. C. Le, J. Oláh, and M. Pakurár. Network interactions of global supply chain members. *Journal of Business Economics and Management*, 22(6):1593–1613, 2021.

[230] A. Ledwoch, H. Yasarcan, and A. Brintrup. The moderating impact of supply network topology on the effectiveness of risk management. *International Journal of Production Economics*, 197:13–26, 2018.

[231] S. M. Lee and M. J. Schniederjans. A multicriteria assignment problem: A goal programming approach. *Interfaces*, 13(4):75–81, 1983.

[232] F. Legillon, A. Liefooghe, and E.-G. Talbi. Cobra: A cooperative coevolutionary algorithm for bi-level optimization. In *2012 IEEE Congress on evolutionary computation*, pages 1–8. IEEE, 2012.

[233] W.-S. Li and C. Clifton. Semint: A tool for identifying attribute correspondences in heterogeneous databases using neural networks. *Data & Knowledge Engineering*, 33(1):49–84, 2000.

[234] W.-S. Li, C. Clifton, and S.-Y. Liu. Database integration using neural networks: implementation and experiences. *Knowledge and Information Systems*, 2(1):73–96, 2000.

[235] Y. Li. An automatic virtual organization structure modeling method in supply chain management. In *2010 International Conference on Management and Service Science*, pages 1–4. IEEE, 2010.

[236] Y. Li and Z. Chen. The distributed permutation flowshop scheduling problem: A genetic algorithm approach. In *2015 3rd International Conference on Mechatronics and Industrial Informatics (ICMII 2015)*, pages 381–384. Atlantis Press, 2015.

[237] Y. Li, D.-B. Liu, and W.-M. Zhang. Schema matching using neural network. In *The 2005 IEEE/WIC/ACM International Conference on Web Intelligence (WI'05)*, pages 743–746. IEEE, 2005.

[238] Y. Liang. An instance-based approach for domain-independent schema matching. In *Proceedings of the 46th Annual Southeast Regional Conference on XX*, pages 268–271. ACM, 2008.

[239] C. Lin, K. L. Choy, G. T. Ho, S. H. Chung, and H. Lam. Survey of green vehicle routing problem: past and future trends. *Expert systems with applications*, 41(4):1118–1138, 2014.

[240] C. K. Y. Lin. Solving a location, allocation, and capacity planning problem with dynamic demand and response time service level. *Mathematical Problems in Engineering*, 2014, 2014.

[241] D. Lin et al. An information-theoretic definition of similarity. In *Icml*, volume 98, pages 296–304. Citeseer, 1998.

[242] D.-Y. Lin and H.-Y. Liu. Combined ship allocation, routing and freight assignment in tramp shipping. *Transportation Research Part E: Logistics and Transportation Review*, 47(4):414–431, 2011.

[243] C. Liu, H. Duan, Z. Qingtian, M. Zhou, F. Lu, and J. Cheng. Towards comprehensive support for privacy preservation cross-organization business process mining. *IEEE Transactions on Services Computing*, 2016.

[244] R. Lotfi, Y. Z. Mehrjerdi, M. S. Pishvaee, A. Sadeghieh, and G.-W. Weber. A robust optimization model for sustainable and resilient closed-loop supply chain network design considering conditional value at risk. *Numerical algebra, control & optimization*, 11(2):221, 2021.

[245] H. R. Lourenço, O. C. Martin, and T. Stützle. Iterated local search. In *Handbook of metaheuristics*, pages 320–353. Springer, 2003.

[246] C. Lu, L. Gao, W. Gong, C. Hu, X. Yan, and X. Li. Sustainable scheduling of distributed permutation flow-shop with non-identical factory using a knowledge-based multi-objective memetic optimization algorithm. *Swarm and Evolutionary Computation*, 60:100803, 2021.

[247] C. Lu, Y. Huang, L. Meng, L. Gao, B. Zhang, and J. Zhou. A pareto-based collaborative multi-objective optimization algorithm for energy-efficient scheduling of distributed permutation flow-shop with limited buffers. *Robotics and Computer-Integrated Manufacturing*, 74:102277, 2022.

[248] G. Luo, X. Wen, H. Li, W. Ming, and G. Xie. An effective multi-objective genetic algorithm based on immune principle and external archive for multi-objective integrated process planning and scheduling. *The International Journal of Advanced Manufacturing Technology*, 91(9):3145–3158, 2017.

[249] J. Madhavan, P. A. Bernstein, and E. Rahm. Generic schema matching with cupid. In *vldb*, volume 1, pages 49–58, 2001.

[250] S. Mahdavi, M. E. Shiri, and S. Rahnamayan. Metaheuristics in large-scale global continues optimization: A survey. *Information Sciences*, 295:407–428, 2015.

[251] S. Mandal. A social-exchange perspective on supply chain innovation. *International Journal of Information Systems in the Service Sector (IJISSS)*, 8(3):36–57, 2016.

[252] J. A. Manson, T. W. Chamberlain, and R. A. Bourne. Mvmoo: Mixed variable multi-objective optimisation. *Journal of Global Optimization*, 80(4):865–886, 2021.

[253] I. Manuj and F. Sahin. A model of supply chain and supply chain decision-making complexity. *International Journal of Physical Distribution & Logistics Management*, 2011.

[254] M. Maric. An efficient genetic algorithm for solving the multi-level uncapacitated facility location problem. *Computing and Informatics*, 29(2):183, 2010.

[255] A. Marie and A. Gal. Managing uncertainty in schema matcher ensembles. In *International Conference on Scalable Uncertainty Management*, pages 60–73. Springer, 2007.

[256] S. Martello, W. R. Pulleyblank, P. Toth, and D. De Werra. Balanced optimization problems. *Operations Research Letters*, 3(5):275–278, 1984.

[257] L. Maruster, J. H. Wortmann, A. T. Weijters, and W. W. van der Aalst. Discovering distributed processes in supply chains. In *Collaborative Systems for Production Management*, pages 219–230. Springer, 2003.

[258] M. Marzouk and M. Sabbah. Ahp-topsis social sustainability approach for selecting supplier in construction supply chain. *Cleaner Environmental Systems*, 2:100034, 2021.

[259] N. Matinrad, E. Roghanian, and Z. Rasi. Supply chain network optimization: A review of classification, models, solution techniques and future research. *Uncertain Supply Chain Management*, 1(1):1–24, 2013.

[260] J. Mazzola and A. Neebe. Bottleneck generalized assignment problems. *Engineering Costs and Production Economics*, 14(1):61–65, 1988.

[261] J. B. Mazzola and A. W. Neebe. Resource-constrained assignment scheduling. *Operations Research*, 34(4):560–572, 1986.

[262] O. A. Mehdi, H. Ibrahim, and L. S. Affendey. Instance based matching using regular expression. *Procedia Computer Science*, 10:688–695, 2012.

[263] O. A. Mehdi, H. Ibrahim, and L. S. Affendey. An approach for instance based schema matching with google similarity and regular expression. *Int. Arab J. Inf. Technol.*, 14(5):755–763, 2017.

[264] P. Melin, L. Astudillo, O. Castillo, F. Valdez, and M. Garcia. Optimal design of type-2 and type-1 fuzzy tracking controllers for autonomous mobile robots under perturbed torques using a new chemical optimization paradigm. *Expert Systems with Applications*, 40(8):3185–3195, 2013.

[265] M. T. Melo, S. Nickel, and F. Saldanha-Da-Gama. Facility location and supply chain management–a review. *European journal of operational research*, 196(2):401–412, 2009.

[266] Z. Michalewicz. Quo vadis, evolutionary computation? In *Advances in Computational Intelligence*, pages 98–121. Springer, 2012.

[267] K. Miettinen. *Nonlinear multiobjective optimization*, volume 12. Springer Science & Business Media, 2012.

[268] K. Miettinen and M. Mäkelä. Interactive bundle-based method for nondifferentiable multiobjeective optimization: Nimbus. *Optimization*, 34(3):231–246, 1995.

[269] R. Mihalcea, C. Corley, C. Strapparava, et al. Corpus-based and knowledge-based measures of text semantic similarity. In *Aaai*, volume 6, pages 775–780, 2006.

[270] G. A. Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

[271] S. Mirjalili. Sca: a sine cosine algorithm for solving optimization problems. *Knowledge-based systems*, 96:120–133, 2016.

[272] S. Mirzapour Al-E-Hashem, H. Malekly, and M. B. Aryanezhad. A multi-objective robust optimization model for multi-product multi-site aggregate production planning in a supply chain under uncertainty. *International Journal of Production Economics*, 134(1):28–42, 2011.

[273] N. Mladenović and P. Hansen. Variable neighborhood search. *Computers & operations research*, 24(11):1097–1100, 1997.

[274] S. Moons, K. Ramaekers, A. Caris, and Y. Arda. Integrating production scheduling and vehicle routing decisions at the operational decision level: a review and discussion. *Computers & Industrial Engineering*, 104:224–245, 2017.

[275] N. Moosavian, B. K. Roodsari, et al. Soccer league competition algorithm, a new method for solving systems of nonlinear equations. *International journal of intelligence science*, 4(01):7, 2013.

[276] R. A. Mora-Gutiérrez, J. Ramírez-Rodríguez, E. A. Rincón-García, A. Ponsich, and O. Herrera. An optimization algorithm inspired by social creativity systems. *Computing*, 94(11):887–914, 2012.

[277] P. K. Muhuri, Z. Ashraf, and Q. D. Lohani. Multiobjective reliability redundancy allocation problem with interval type-2 fuzzy uncertainty. *IEEE Transactions on Fuzzy Systems*, 26(3):1339–1355, 2017.

[278] S. Munir, F. Khan, and M. A. Riaz. An instance based schema matching between opaque database schemas. In *2014 4th International Conference on Engineering Technology and Technopreneuship (ICE2T)*, pages 177–182. IEEE, 2014.

[279] B. Naderi and R. Ruiz. The distributed permutation flowshop scheduling problem. *Computers & Operations Research*, 37(4):754–768, 2010.

[280] A. Nandi and P. A. Bernstein. Hamster: using search clicklogs for schema and taxonomy matching. *Proceedings of the VLDB Endowment*, 2(1):181–192, 2009.

[281] S. Nayak. *Fundamentals of Optimization Techniques with Algorithms*. Academic Press, 2020.

[282] T. T. Nguyen, Z. Yang, and S. Bonsall. Dynamic time-linkage problems-the challenges. In *2012 IEEE RIVF International Conference on Computing & Communication Technologies, Research, Innovation, and Vision for the Future*, pages 1–6. IEEE, 2012.

[283] R. Oftadeh, M. Mahjoob, and M. Shariatpanahi. A novel meta-heuristic optimization algorithm inspired by group hunting of animals: Hunting search. *Computers & Mathematics with Applications*, 60(7):2087–2098, 2010.

[284] Y.-S. Ong and A. Gupta. Evolutionary multitasking: a computer science view of cognitive multitasking. *Cognitive Computation*, 8(2):125–142, 2016.

[285] S. Opricovic. Multicriteria optimization of civil engineering systems. *Faculty of civil engineering, Belgrade*, 2(1):5–21, 1998.

[286] E. Osaba, F. Diaz, and E. Onieva. Golden ball: a novel meta-heuristic to solve combinatorial optimization problems based on soccer concepts. *Applied Intelligence*, 41(1):145–166, 2014.

[287] E. H. Özder, E. Özcan, and T. Eren. A systematic literature review for personnel scheduling problems. *International Journal of Information Technology & Decision Making*, 19(06):1695–1735, 2020.

[288] G. Pankratz. A grouping genetic algorithm for the pickup and delivery problem with time windows. *Or Spectrum*, 27(1):21–41, 2005.

[289] T. Papenbrock and F. Naumann. A hybrid approach for efficient unique column combination discovery. *Datenbanksysteme für Business, Technologie und Web (BTW 2017)*, 2017.

[290] M. Park, M. Song, T. H. Baek, S. Son, S. J. Ha, and S. W. Cho. Workload and delay analysis in manufacturing process using process mining. In *Asia-Pacific Conference on Business Process Management*, pages 138–151. Springer, 2015.

[291] A. Pascoletti and P. Serafini. Scalarizing vector optimization problems. *Journal of Optimization Theory and Applications*, 42(4):499–524, 1984.

[292] A. J. Peirleitner, K. Altendorfer, and T. Felberbauer. A simulation approach for multistage supply chain optimization to analyze real world transportation effects. In *2016 Winter Simulation Conference (WSC)*, pages 2272–2283. IEEE, 2016.

[293] D. W. Pentico. Assignment problems: A golden anniversary survey. *European Journal of Operational Research*, 176(2):774–793, 2007.

[294] E. Perea-Lopez, B. E. Ydstie, and I. E. Grossmann. A model predictive control strategy for supply chain optimization. *Computers & Chemical Engineering*, 27(8-9):1201–1218, 2003.

[295] H. L. Petersen, C. Archetti, and M. G. Speranza. Exact solutions to the double travelling salesman problem with multiple stacks. *Networks*, 56(4):229–243, 2010.

[296] C. Pike-Burke. Multi-objective optimization, 2019.

[297] V. Pillac, M. Gendreau, C. Guéret, and A. L. Medaglia. A review of dynamic vehicle routing problems. *European Journal of Operational Research*, 225(1):1–11, 2013.

[298] M. Pinedo. *Planning and scheduling in manufacturing and services*. Springer, 2005.

[299] D. Pinto, D. Vilarino, Y. Alemán, H. Gómez, and N. Loya. The soundex phonetic algorithm revisited for sms-based information retrieval. In *II Spanish Conference on Information Retrieval CERI*, 2012.

[300] M. A. Potter and K. A. D. Jong. Cooperative coevolution: An architecture for evolving coadapted subcomponents. *Evolutionary computation*, 8(1):1–29, 2000.

[301] A. P. Punnen and Y. Aneja. Categorized assignment scheduling: A tabu search approach. *Journal of the Operational Research Society*, 44(7):673–679, 1993.

[302] P. Rabanal, I. Rodríguez, and F. Rubio. Using river formation dynamics to design heuristic algorithms. In *International conference on unconventional computation*, pages 163–177. Springer, 2007.

[303] L. Rabelo, M. Helal, and C. Lertpattarapong. Analysis of supply chains using system dynamics, neural nets, and eigenvalues. In *Proceedings of the 2004 Winter Simulation Conference, 2004.*, volume 2, pages 1136–1144. IEEE, 2004.

[304] R. Rahim, S. Supiyandi, A. Siahaan, T. Listyorini, A. P. Utomo, W. A. Triyanto, Y. Irawan, S. Aisyah, M. Khairani, S. Sundari, et al. Topsis method application for decision support system in internal control for selecting best employees. In *Journal of Physics: Conference Series*, volume 1028, page 012052. IOP Publishing, 2018.

[305] E. Rahm. Towards large-scale schema and ontology matching. In *Schema matching and mapping*, pages 3–27. Springer, 2011.

[306] E. Rahm and P. A. Bernstein. A survey of approaches to automatic schema matching. *the VLDB Journal*, 10(4):334–350, 2001.

[307] R. V. Rao, V. J. Savsani, and D. Vakharia. Teaching–learning-based optimization: a novel method for constrained mechanical design optimization problems. *Computer-aided design*, 43(3):303–315, 2011.

[308] S. S. Rao. Game theory approach for multiobjective structural optimization. *Computers & Structures*, 25(1):119–127, 1987.

[309] E. Rashedi, H. Nezamabadi-Pour, and S. Saryazdi. Gsa: a gravitational search algorithm. *Information sciences*, 179(13):2232–2248, 2009.

[310] L. Ratinov and E. Gudes. Abbreviation expansion in schema matching and web integration. In *Proceedings of the 2004 IEEE/WIC/ACM International Conference on Web Intelligence*, pages 485–489. IEEE Computer Society, 2004.

[311] I. Rechenberg. Evolution strategy: Nature's way of optimization. In *Optimization: Methods and applications, possibilities and limitations*, pages 106–126. Springer, 1989.

[312] M. Reimann, R. Tavares Neto, and E. Bogendorfer. Joint optimization of production planning and vehicle routing problems: A review of existing strategies. *Pesquisa Operacional*, 34(2):189–214, 2014.

[313] G. Reinelt. Tsplib95. *Interdisziplinäres Zentrum für Wissenschaftliches Rechnen (IWR), Heidelberg*, 338:1–16, 1995.

[314] R. G. Reynolds. An introduction to cultural algorithms. In *Proceedings of the third annual conference on evolutionary programming*, volume 24, pages 131–139. World Scientific, 1994.

[315] A. Ridwan, A. Bahauddin, and R. Naufal. Optimization of supply chain operation cost and gas usage quantity using non-dominated sorting genetic algorithm ii (nsga-ii) method. In *IOP Conference Series: Materials Science and Engineering*, volume 909, page 012063. IOP Publishing, 2020.

[316] J. O. Robles, C. Azzaro-Pantel, and A. Aguilar-Lasserre. Optimization of a hydrogen supply chain network design under demand uncertainty by multi-objective genetic algorithms. *Computers & Chemical Engineering*, 140:106853, 2020.

[317] S. Rong, X. Niu, E. W. Xiang, H. Wang, Q. Yang, and Y. Yu. A machine learning approach for instance matching based on similarity metrics. In *International Semantic Web Conference*, pages 460–475. Springer, 2012.

[318] R. E. Rosenthal. Concepts, theory, and techniques principles of multiobjective optimization. *Decision Sciences*, 16(2):133–152, 1985.

[319] A. Rostin, O. Albrecht, J. Bauckmann, F. Naumann, and U. Leser. A machine learning approach to foreign key discovery. In *WebDB*, 2009.

[320] B. Roy. Classement et choix en présence de points de vue multiples. *Revue française d'informatique et de recherche opérationnelle*, 2(8):57–75, 1968.

[321] S. Rozsnyai, G. T. Lakshmanan, V. Muthusamy, R. Khalaf, and M. J. Duftler. Business process insight: An approach and platform for the discovery and analysis of end-to-end business processes. In *2012 Annual SRII Global Conference*, pages 80–89. IEEE, 2012.

[322] R. Y. Rubinstein. Optimization of computer simulation models with rare events. *European Journal of Operational Research*, 99(1):89–112, 1997.

[323] C. Ryan, J. J. Collins, and M. O. Neill. Grammatical evolution: Evolving programs for an arbitrary language. In *European conference on genetic programming*, pages 83–96. Springer, 1998.

[324] T. L. Saaty. The analytic hierarchy process mcgraw-hill. *New York*, 324, 1980.

[325] L. Saglietto, F. Fulconis, D. Bédé, and J. de Almeira Goes. Wine industry supply chain (wsc) modeling: an argentina-france comparison. In *Third international workshop on food supply chain*, 2014.

[326] S. A. Salem. Boa: A novel optimization algorithm. In *2012 international conference on engineering and technology (ICET)*, pages 1–5. IEEE, 2012.

[327] M. H. Salmani and K. Eshghi. A metaheuristic algorithm based on chemotherapy science: Csa. *Journal of Optimization*, 2017, 2017.

[328] G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. *Information processing & management*, 24(5):513–523, 1988.

[329] V. Sathiya, M. Chinnadurai, S. Ramabalan, and A. Appolloni. Mobile robots and evolutionary optimization algorithms for green supply chain management in a used-car resale company. *Environment, Development and Sustainability*, 23(6):9110–9138, 2021.

[330] P. E. Seiden and F. Celada. A model for simulating cognate recognition and response in the immune system. *Journal of theoretical biology*, 158(3):329–357, 1992.

[331] G. Seliger, B. Viehweger, and B. Wieneke. Decision support in design and optimization of flexible automated manufacturing and assembly. *Robotics and computer-integrated manufacturing*, 3(2):221–227, 1987.

[332] S. Serdarasan and M. Tanyas. Dealing with complexity in the supply chain: The effect of supply chain management initiatives. *Available at SSRN 2056331*, 2012.

[333] M. Shigeno, Y. Saruwatari, and T. Matsui. An algorithm for fractional assignment problems. *Discrete Applied Mathematics*, 56(2-3):333–343, 1995.

[334] A. Shoja, S. Molla-Alizadeh-Zavardehi, and S. Niroomand. Hybrid adaptive simplified human learning optimization algorithms for supply chain network design problem with possibility of direct shipment. *Applied Soft Computing*, 96:106594, 2020.

[335] A. Shtub and K. Kogan. Capacity planning by the dynamic multi-resource generalized assignment problem (dmrgap). *European Journal of Operational Research*, 105(1):91–99, 1998.

[336] Y. Shuai, S. Yunfeng, and Z. Kai. An effective method for solving multiple travelling salesman problem based on nsga-ii. *Systems Science & Control Engineering*, 7(2):108–116, 2019.

[337] H. A. Simon. The architecture of complexity. *Proc Am Philos Soc*, 106:467–482, 1962.

[338] L. V. Snyder. Facility location under uncertainty: a review. *IIE transactions*, 38(7):547–564, 2006.

[339] J. D. Sterman. *Business dynamics: systems thinking and modeling for a complex world*. Number HD30. 2 S7835 2000. 2000.

[340] W. Stewart. The delivery truck routing problem with stochastic demands. In *ORSA/TIMS meeting*, 1976.

[341] J. Stolk, I. Mann, A. Mohais, and Z. Michalewicz. Combining vehicle routing and packing for optimal delivery schedules of water tanks. *OR Insight*, 26(3):167–190, 2013.

[342] S. A. Suhi, R. Enayet, T. Haque, S. M. Ali, M. A. Moktadir, and S. K. Paul. Environmental sustainability assessment in supply chain: an emerging economy context. *Environmental Impact Assessment Review*, 79:106306, 2019.

[343] E. Taillard. Benchmarks for basic scheduling problems. *european journal of operational research*, 64(2):278–285, 1993.

[344] É. D. Taillard and S. Voss. Popmusic—partial optimization metaheuristic under special intensification conditions. In *Essays and surveys in metaheuristics*, pages 613–629. Springer, 2002.

[345] A. A. Tako and S. Robinson. The application of discrete event simulation and system dynamics in the logistics and supply chain context. *Decision support systems*, 52(4):802–815, 2012.

[346] E.-G. Talbi. A taxonomy of metaheuristics for bi-level optimization. In *Metaheuristics for bi-level optimization*, pages 1–39. Springer, 2013.

[347] Y. Tan and Y. Zhu. Fireworks algorithm for optimization. In *International conference in swarm intelligence*, pages 355–364. Springer, 2010.

[348] Z. Tang and M. Gong. Adaptive multifactorial particle swarm optimisation. *CAAI Transactions on Intelligence Technology*, 4(1):37–46, 2019.

[349] E. Tanyildizi and G. Demir. Golden sine algorithm: A novel math-inspired algorithm. *Advances in Electrical and Computer Engineering*, 17(2):71–78, 2017.

[350] Z. W. Thu, D. Kim, J. Lee, W.-J. Won, H. J. Lee, N. L. Ywet, A. A. Maw, and J.-W. Lee. Multivehicle point-to-point network problem formulation for uam operation management used with dynamic scheduling. *Applied Sciences*, 12(22):11858, 2022.

[351] J. Timmis. *Artificial immune systems: A novel data analysis technique inspired by the immune network theory*. PhD thesis, Department of Computer Science, 2000.

[352] H. Tohyama, K. Ida, and J. Matsueda. A genetic algorithm for the uncapacitated facility location problem. *Electronics and Communications in Japan*, 94(5):47–54, 2011.

[353] E. Triantaphyllou, B. Shu, S. N. Sanchez, and T. Ray. Multi-criteria decision making: an operations research approach. *Encyclopedia of electrical and electronics engineering*, 15(1998):175–186, 1998.

[354] T. Trisna, M. Marimin, Y. Arkeman, and T. Sunarti. Multi-objective optimization for supply chain management problem: A literature review. *Decision Science Letters*, 5(2):283–316, 2016.

[355] T. H. Truong and F. Azadivar. Simulation optimization in manufacturing analysis: simulation based optimization for supply chain configuration design. In *Proceedings of the 35th conference on Winter simulation: driving innovation*, pages 1268–1275. Winter Simulation Conference, 2003.

[356] C. A. Ullrich. Integrated machine scheduling and vehicle routing with time windows. *European Journal of Operational Research*, 227(1):152–165, 2013.

[357] W. Van Der Aalst, A. Adriansyah, A. K. A. De Medeiros, F. Arcieri, T. Baier, T. Blickle, J. C. Bose, P. Van Den Brand, R. Brandtjen, J. Buijs, et al. Process mining manifesto. In *International Conference on Business Process Management*, pages 169–194. Springer, 2011.

[358] A. Van Horenbeek, J. Buré, D. Cattrysse, L. Pintelon, and P. Vansteenwegen. Joint maintenance and inventory optimization systems: A review. *International Journal of Production Economics*, 143(2):499–508, 2013.

[359] F. J. Varela and A. Coutinho. Second generation immune networks. *Immunology today*, 12(5):159–166, 1991.

[360] V. Vennila, S. Savitha, and T. Sathya. Record linkage using jaro-winkler technique implemented in database misuse domain. *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)*, 4(3), 2015.

[361] V. S. Verykios, A. K. Elmagarmid, and E. N. Houstis. Automating the approximate record-matching process. *Information sciences*, 126(1-4):83–98, 2000.

[362] T. Vicsek, A. Czirók, E. Ben-Jacob, I. Cohen, and O. Shochet. Novel type of phase transition in a system of self-driven particles. *Physical review letters*, 75(6):1226, 1995.

[363] D. K. Vieira, G. L. Soares, J. A. Vasconcelos, and M. H. Mendes. A genetic algorithm for multi-component optimization problems: the case of the travelling thief problem. In *European Conference on Evolutionary Computation in Combinatorial Optimization*, pages 18–29. Springer, 2017.

[364] A. Volgenant. Linear and semi-assignment problems: a core oriented approach. *Computers & Operations Research*, 23(10):917–932, 1996.

[365] C. Voudouris, G. Owusu, R. Dorne, C. Ladde, and B. Virginas. Arms: An automated resource management system for british telecommunications plc. *European journal of operational research*, 171(3):951–961, 2006.

[366] C. Voudouris and E. Tsang. Function optimization using guided local search. *University of Essex, Technical Report CSM-249, Colchester, UK*, 1995.

[367] R. A. Walker and S. Chaudhuri. Introduction to the scheduling problem. *IEEE Design & Test of Computers*, 12(2):60–69, 1995.

[368] C. Wang. Research on optimal design of short-life cycle product logistics supply chain based on multicriteria decision model. *Security and Communication Networks*, 2021, 2021.

[369] G. Wang, L. Gao, X. Li, P. Li, and M. F. Tasgetiren. Energy-efficient distributed permutation flow shop scheduling problem using a multi-objective whale swarm algorithm. *Swarm and Evolutionary Computation*, 57:100716, 2020.

[370] J. Wang and S. E. Madnick. The inter-database instance identification problem in integrating autonomous systems. In *[1989] Proceedings. Fifth International Conference on Data Engineering*, pages 46–55. IEEE, 1989.

[371] X. Wang and S. M. Disney. The bullwhip effect: Progress, trends and directions. *European Journal of Operational Research*, 250(3):691–701, 2016.

[372] Y. Wang, J. Qin, and W. Wang. Efficient approximate entity matching using jaro-winkler distance. In *International Conference on Web Information Systems Engineering*, pages 231–239. Springer, 2017.

[373] X. Wen, X. Li, L. Gao, K. Wang, and H. Li. Modified honey bees mating optimization algorithm for multi-objective uncertain integrated process planning and scheduling problem. *International Journal of Advanced Robotic Systems*, 17(3):1729881420925236, 2020.

[374] W. E. Winkler. Frequency-based matching in fellegi-sunter model of record linkage. *Bureau of the Census Statistical Research Division*, 14, 2000.

[375] F. Xie, C. N. Potts, and T. Bektaş. Iterated local search for workforce scheduling and routing problems. *Journal of Heuristics*, 23(6):471–500, 2017.

[376] J. Xu and J. Zhang. Exploration-exploitation tradeoffs in metaheuristics: Survey and analysis. In *Proceedings of the 33rd Chinese control conference*, pages 8633–8638. IEEE, 2014.

[377] Y. Xu, Z. Cui, and J. Zeng. Social emotional optimization algorithm for nonlinear constrained optimization problems. In *International conference on swarm, evolutionary, and memetic computing*, pages 583–590. Springer, 2010.

[378] A. M. Yaakob and A. Gegov. Interactive topsis based group decision making methodology using z-numbers. *International Journal of Computational Intelligence Systems*, 9(2):311–324, 2016.

[379] X.-S. Yang. A new metaheuristic bat-inspired algorithm. *Nature inspired cooperative strategies for optimization (NICSO 2010)*, pages 65–74, 2010.

[380] X.-S. Yang and S. Deb. Engineering optimisation by cuckoo search. *International Journal of Mathematical Modelling and Numerical Optimisation*, 1(4):330–343, 2010.

[381] Y. Yang, M. Chen, and B. Gao. An effective content-based schema matching algorithm. In *2008 International Seminar on Future Information Technology and Management Engineering*, pages 7–11. IEEE, 2008.

[382] Z. Yang, K. Tang, and X. Yao. Large scale evolutionary optimization using cooperative coevolution. *Information sciences*, 178(15):2985–2999, 2008.

[383] H. Yano and M. Sakawa. A unified approach for characterizing pareto optimal solutions of multiobjective optimization problems: The hyperplane method. *European Journal of Operational Research*, 39(1):61–70, 1989.

[384] F. E. Yates. Complexity and the limits to knowledge. *American Journal of Physiology-Regulatory, Integrative and Comparative Physiology*, 235(5):R201–R204, 1978.

[385] P.-L. Yu. A class of solutions for group decision problems. *Management science*, 19(8):936–946, 1973.

[386] S. Yu, Z. Han, and J. Le. A flexible and composite schema matching algorithm. In *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"*, pages 55–65. Springer, 2004.

[387] Y. Yuan, Y.-S. Ong, A. Gupta, P. S. Tan, and H. Xu. Evolutionary multitasking in permutation-based combinatorial optimization problems: Realization with tsp, qap, lop, and jsp. In *2016 IEEE Region 10 Conference (TENCON)*, pages 3157–3164. IEEE, 2016.

[388] B. Zapilko, M. Zloch, and J. Schaible. Utilizing regular expressions for instance-based schema matching. In *Proceedings of the 7th International Conference on Ontology Matching-Volume 946*, pages 240–241. CEUR-WS. org, 2012.

[389] S. H. Zegordi and M. A. B. Nia. Integrating production and transportation scheduling in a two-stage supply chain considering order assignment. *The International Journal of Advanced Manufacturing Technology*, 44(9-10):928–939, 2009.

[390] P. Zelbst, K. W. Green Jr, V. E. Sower, and P. Reyes. Supply chain linkages: power, benefits, and risk reduction. In *Annual Meeting of the Southwest Decision Sciences Institute, Oklahoma City 2009*, pages 483–496, 2009.

[391] P. J. Zelbst, K. W. Green Jr, V. E. Sower, and P. Reyes. Impact of supply chain linkages on supply chain performance. *Industrial Management & Data Systems*, 109(5):665–682, 2009.

[392] A. Zemzam, M. E. Maataoui, M. Hlyal, J. E. Alami, and N. E. Alami. Inventory management of supply chain with robust control theory: literature review. *International Journal of Logistics Systems and Management*, 27(4):438–465, 2017.

[393] Q. Zeng, S. X. Sun, H. Duan, C. Liu, and H. Wang. Cross-organizational collaborative workflow mining from a multi-source log. *Decision support systems*, 54(3):1280–1301, 2013.

[394] P. Zerbino, A. Stefanini, and D. Aloini. Process science in action: A literature review on process mining in business management. *Technological Forecasting and Social Change*, 172:121021, 2021.

[395] L. Zhang, Z. Liu, W. Wang, and B. Yu. Long-term charging infrastructure deployment and bus fleet transition considering seasonal differences. *Transportation Research Part D: Transport and Environment*, 111:103429, 2022.

[396] M. Zhang, M. Hadjieleftheriou, B. C. Ooi, C. M. Procopiuc, and D. Srivastava. On multi-column foreign key discovery. *Proceedings of the VLDB Endowment*, 3(1-2):805–814, 2010.

[397] M. Zhang, M. Hadjieleftheriou, B. C. Ooi, C. M. Procopiuc, and D. Srivastava. Automatic discovery of attributes in relational databases. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data*, pages 109–120. ACM, 2011.

[398] T. Zhang, W. Gruver, and M. H. Smith. Team scheduling by genetic search. In *Proceedings of the Second International Conference on Intelligent Processing and Manufacturing of Materials. IPMM'99 (Cat. No. 99EX296)*, volume 2, pages 839–844. IEEE, 1999.

[399] X. Zhang, X.-T. Li, and M.-H. Yin. An enhanced genetic algorithm for the distributed assembly permutation flowshop scheduling problem. *International Journal of Bio-Inspired Computation*, 15(2):113–124, 2020.

[400] Z. Zhang and J. Xu. Applying nonlinear modm model to supply chain management with quantity discount policy under complex fuzzy environment. *Journal of Industrial Engineering and Management*, 7(3):660–680, 2014.

[401] H. Zhao and S. Ram. Combining schema and instance information for integrating heterogeneous data sources. *Data & Knowledge Engineering*, 61(2):281–303, 2007.

[402] L. Zhou and X. Xu. Engineering value chain modelling and optimization. In *Value Creation through Engineering Excellence*, pages 205–230. Springer, 2018.

[403] Z. Zhou, S. Cheng, and B. Hua. Supply chain optimization of continuous process industries with sustainability considerations. *Computers & Chemical Engineering*, 24(2-7):1151–1158, 2000.

[404] E. Zitzler and L. Thiele. Multiobjective optimization using evolutionary algorithms—a comparative case study. In *International conference on parallel problem solving from nature*, pages 292–301. Springer, 1998.

[405] E. Zitzler and L. Thiele. Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. *IEEE transactions on Evolutionary Computation*, 3(4):257–271, 1999.

[406] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. G. Da Fonseca. Performance assessment of multiobjective optimizers: An analysis and review. *IEEE Transactions on evolutionary computation*, 7(2):117–132, 2003.

[407] G. Zobolas, C. D. Tarantilis, and G. Ioannou. Minimizing makespan in permutation flow shop scheduling problems using a hybrid metaheuristic algorithm. *Computers & Operations Research*, 36(4):1249–1267, 2009.

[408] I. Zouaghi, A. Spalanzani, and A. Gunasekaran. Impact of supply chain relational interactions on information sharing. 2011.