Distributed Decision-making in Disrupted Industrial Environments Using a Multi-agent Framework

by

Mingjie Bi

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Robotics)
in the University of Michigan
2023

Doctoral Committee:

Professor Kira Barton, Co-Chair
Professor Dawn Tilbury, Co-Chair
Professor Stéphane Lafortune
Professor Siqian Shen
Assistant Professor Vasileios Tzoumas

Mingjie Bi
mingjieb@umich.edu
ORCID iD: 0000-0001-6825-1445

ACKNOWLEDGMENTS

Throughout my journey of studying in the U.S., I have been supported and encouraged by numerous people who played a role in completing my academic achievements, particularly in completing this Ph.D. dissertation. Foremost, I am deeply grateful to my esteemed advisors, Professor Kira Barton and Professor Dawn M. Tilbury, who provided me with the invaluable opportunity to pursue a doctoral degree and financial support during my time at the university. They have been extremely supportive of my academic performance and daily life wellness. Their dedication to academic excellence and exceptional mentorship enables me to grow and shape my technical and personal skills as a researcher. The understanding and patience they expressed during the challenging periods have been a constant source of strength, enabling me to persist and thrive. My graduate school experience has been greatly enriched by their guidance and support. I would also like to extend my appreciation to my committee members, Dr. Siqian Shen, Dr. Stéphane Lafortune, and Dr. Vasileios Tzoumas. Their expertise and willingness to review my work have been invaluable. Their constructive feedback and insights have significantly enhanced the quality of this dissertation.

In addition, I would like to express my gratitude to every member of Barton Research Group and Tilbury Research Group. I am deeply appreciative of the enriching exchange of ideas, collaborative efforts, and constructive debates with colleagues such as Ilya Kovalenko, Max Wu, Loubna Baroudi, Bo Fu, Tylor Toner, Efe Balta, Issac Spiegel, and Yassine Qamsane, among others. Their contributions have greatly deepened my understanding of the subject matter and motivated me to explore new perspectives. I would also like to thank Max Toothman, Nazanin Farjam, Kaifan Yue, Angelo Hawa, Ali Bahrami, Keith Ng, and Leo Bringer to enrich my daily life and foster a sense of belonging in Ann Arbor. Outside of the research group, I owe great appreciation to my friends and cohorts at the University of Michigan, including Pinyi Wu, Hongyu Zhou, Yaohong Xiao, Xinxin Yao, Liangguang Lin, Haonan Chang, Zekun Bian, Xi Lin, Eva Mungai, etc., for their unwavering support in both my academic pursuits and personal well-being.

I am immensely grateful to my friends and family in China for their support and encouragement throughout my life in the United States. I am truly blessed to have such a caring and supportive network of my childhood friends, Tianhao Dong, Gongqiang Huang, Mohan Zhang, Chenxi Liu, Zixing Guo, Yiqun Wang, as well as friends from undergrad, Xiang Su, Yu Dong, Yangliuer Chen, and Yifei Yu. Their faith in me has inspired me to work harder and strive for excellence in my studies. I would also like to thank my big family, especially my cousins, Yongzhen Wang, Yongqi Yang, and Jiaqi Bi for the countless late-night conversations filled with love from home, which become a magic cure for my homesickness. Of special significance, I would like to extend my heartfelt gratitude to Xu Sun. Your support during challenging times has been an immense source of strength, and I am profoundly grateful for your presence by my side. My sincerest well-wishes go out to you, and I genuinely hope for nothing but the very best in all your endeavors. Last, but not least, I am heavily indebted to my parents, Jinghuan Bi, and Qingfang Yang, for their unwavering love and encouragement. Their constant belief in my abilities, along with the sacrifices they have made to support my education, have been the bedrock of my determination. I am forever grateful for their guidance and the values they instilled in me, which have shaped my character and propelled me forward.

I would like to acknowledge the project funding provided by NSF and all the support from the University of Michigan, Rackham Graduate School, College of Engineering, Robotics Department, and Department of Mechanical Engineering. In closing, I would like to emphasize that this Ph.D. dissertation is not just the culmination of my efforts but a reflection of the collective contributions of all those mentioned above. Each individual has left an indelible mark on this academic journey, and I am profoundly grateful for their support and encouragement.

TABLE OF CONTENTS

LIST OF FIGURES

FIGURE

LIST OF TABLES

TABLE

LIST OF ACRONYMS

AI Artificial Intelligence

IIoT Industrial Internet of Things

DT Digital Twin

DAI Distributed Artificial Intelligence

RFID Radio Frequency Identification

PA Product Agent

RA Resource Agent

DSS Decision Support System

BDI Belief-Desire-Intention

FSM Finite State Machine

MTBF Mean Time Between Failure

OEM Original Equipment Manufacturer

CNP Contract Net Protocol

FIPA Foundation for Intelligent Physical Agents

MILP Mixed-Integer Linear Programming

SAA Sample Average Approximation

IEC ISO/International Electrotechnical Commission

AGV Autonomous Guided Vehicle

ABSTRACT

The modern industrial environment is becoming more complex and dynamic, due to customized and shifting market demands, highly connected businesses, and frequently upgraded technologies. In such environments, varying uncertainties and disruptions could occur and highly impact the performance of the manufacturing factories and supply chain networks. Conventional centralized decision-making approaches handle disruptions by re-optimizing across the entire system regardless of disruption type and scale, which require significant computational efforts, especially for complex and large-scale systems. Therefore, to stay competitive, industrial enterprises need a dynamic and flexible decision-making method that enables an agile and resilient response to unexpected disruptions.

Enabled by current Artificial Intelligence (AI) techniques, multi-agent control has been proposed to conduct distributed decision-making to provide an agile response to disruptions. A multi-agent system consists of various autonomous agents, which are cyber representations of their associated physical objects and have their own knowledge and goals. Agents communicate and interact with each other to make high-level decisions for their associated physical objects. In different industrial environments, agents could represent different system entities, such as the products and machines in manufacturing systems, or suppliers and customers in supply chains. However, most existing industrial multi-agent systems require prior knowledge of disruptions and predetermined rules and strategies to generate responses, which makes it difficult to handle unexpected disruptions.

Aiming to improve the agility and resiliency of industrial systems, this dissertation develops a model-based multi-agent framework to address risk management within an agile and resilient response to various unexpected industrial disruptions. Specifically, this dissertation focuses on the disruptions that affect the occurrence of pre-scheduled events, such as machine breakdown and loss of suppliers. The proposed multi-agent framework comprises model-based agents, heuristic-based communication, and optimization-based decision-making. The model-based agent architecture enables agents to update their knowledge and local environments dynamically. When agents need to make decisions, they utilize their knowledge as heuristics to guide their communication strategies. Then based on their knowledge and communication information, agents can identify new actions by solving risk-aware optimizations to respond to unexpected disruptions dynamically. The pro-

posed framework is tested in a simulated manufacturing environment and a supply chain instance, showcasing the improved flexibility, agility, and resiliency of the multi-agent systems.

To conclude, this dissertation pushes the fields of distributed decision-making for industrial systems closer to satisfying the requirements of modern industry: **flexibility, agility, and resiliency**, especially for manufacturing systems and supply chain networks. Enterprises could apply this framework to address a disruption quickly by computing a resilient recovery plan to minimize the negative effects. In addition, this dissertation contributes to standardizing the design of system-level decision-making using a multi-agent framework. The proposed methodology to design a multi-agent framework is transferable to other complex systems, such as multi-robot systems and autonomous vehicle teams, that consist of multiple intelligent entities.

CHAPTER 1

Introduction

1.1 Motivation

The manufacturing industry has been and will continue to be an important part of the world economy, contributing 27.59% of the global gross domestic product [1] and providing 23% of the employment [2]. In the last decades, the industrial environment has moved towards a global paradigm, where markets demand highly customized and varying products with short life cycles [3, 4] and individual businesses are involved in a wide and complex supply chain network with multiple relationships [5]. In such an environment, customers have more comprehensive demands, including customized products, prices, and levels of service [5]. The rapidly upgraded technologies also enable more complex products and the continuous introduction of new products [6]. External factors, such as social (e.g., trade regulations) and natural factors (e.g., Covid, natural disasters), make the industrial environment more dynamic and unpredictable [7].

This dynamic and ever-changing environment brings various uncertainties and disruptions and induces vulnerability to the manufacturing factories and supply chain networks [8]. These disruptions could occur at all stages in the product life cycle, such as material supply, production, and customer demand, and at different scales [6, 9, 10]. A small-scale disruption example could be a single-machine breakdown, which degrades factory throughput and requires rescheduling to recover production [11]. On a large scale, global events, such as pandemics and wars, could hit the worldwide supply chain network at an unprecedented speed and variety. In the past few years, customers struggled to purchase toilet paper and exhibited increased demand for fitness equipment and office supplies for life at home [12]. Manufacturers needed to reconfigure their production lines to satisfy everyday items and newly required medical equipment, such as ventilators [13]. Suppliers lost delivery channels due to travel and trade regulations. They had to discard products, such as dumping milk, smashing eggs, and wasting raw materials that were no longer needed [14]. The time duration of the disruption and the impact are various and unpredictable. For example, the factories near Fukushima nuclear plant may never come back to use due to the long-term effect, while a fire in a factory could be extinguished in hours, and the production could be recovered in days or weeks [15, 16].

To survive in such environments, enterprises have to take actions, such as reconfiguring the production lines and certifying new suppliers, to cope with disruptions. One research area focusing on disruption response is the decision-making for the rescheduling/re-planning of the system to recover the system performance [11, 17]. Specifically, this dissertation considers the following problem formulation: given an industrial system, existing schedules/plans, and an unexpected disruption, how can we design a decision-making strategy that enables a flexible, agile, and resilient response to unexpected disruption in complex and uncertain industrial environments?

Existing system-level decision-making strategies for adapting to disruptions in manufacturing systems and supply chains are primarily centralized [18, 19]. A centralized approach has a global view of the system and generates a disruption response by utilizing all of the information in the entire system [3, 19, 20]. For the scheduling and planning problems in manufacturing systems and supply chains, researchers have applied different centralized decision-making strategies, such as conventional mathematical programming [21–23] and AI-based learning approaches [24–27]. By modeling the entire system and utilizing big data, these centralized approaches provide optimal solutions based on specific objectives (e.g. cost, throughput), while they generally require significant computational efforts to calculate [3, 19, 20]. Therefore, centralized approaches could benefit the initial scheduling and planning problem and the response to large-scale and long-term disruptions, such as the Fukushima nuclear accident. However, for rescheduling and re-planning after disruptions, all the information about the entire system is required to re-optimize the system in response to the disruption [28]. The time and monetary costs may outweigh the benefits for some small-scale and short-term disruptions. For instance, if a supplier's factory accidentally catches fire and stops supply for days, re-planning the entire supply chain probably needs more time and effort than checking nearby suppliers or just waiting. In conclusion, using centralized approaches limits the agility and responsiveness of the system to handle dynamic environments, and as the complexity and scale of a system increase, it becomes more difficult to effectively manage the whole system under disruptions using centralized methods [6, 29–31].

Dynamic environments impose new requirements on modern industry, namely in terms of **flexibility, agility, and resiliency** [9, 19, 32, 33]. Building upon the literature, the following requirements are defined in the context of disruption response in this dissertation:

- Flexibility is the ability to respond to various disruptions dynamically;
- Agility is the ability to respond to disruptions quickly;
- Resiliency is the ability to maintain system performance against uncertainties after responding to disruptions

To stay competitive, industry enterprises need a dynamic and flexible decision-making method that enables an agile and resilient response to unexpected disruptions in industrial environments [34–36].

The rapid growth of AI techniques, such as machine learning, enables centralized approaches to build prior knowledge (i.e., database) of disruptions and mitigation plans to speed up the response or predict disruptions [37–40]. These approaches rely on an ability to predict disruption accurately and are not flexible for handling unexpected disruptions [7]. On the other hand, AI also enables automated reasoning and knowledge representation at the individual level in systems. Combining the new industrial digital technologies, such as Industrial Internet of Things (IIoT) [41] and Digital Twin (DT) technology [42], industrial systems could move towards individual intelligence [43]. Consequently, distributed decision-making approaches, where multiple entities in the system use

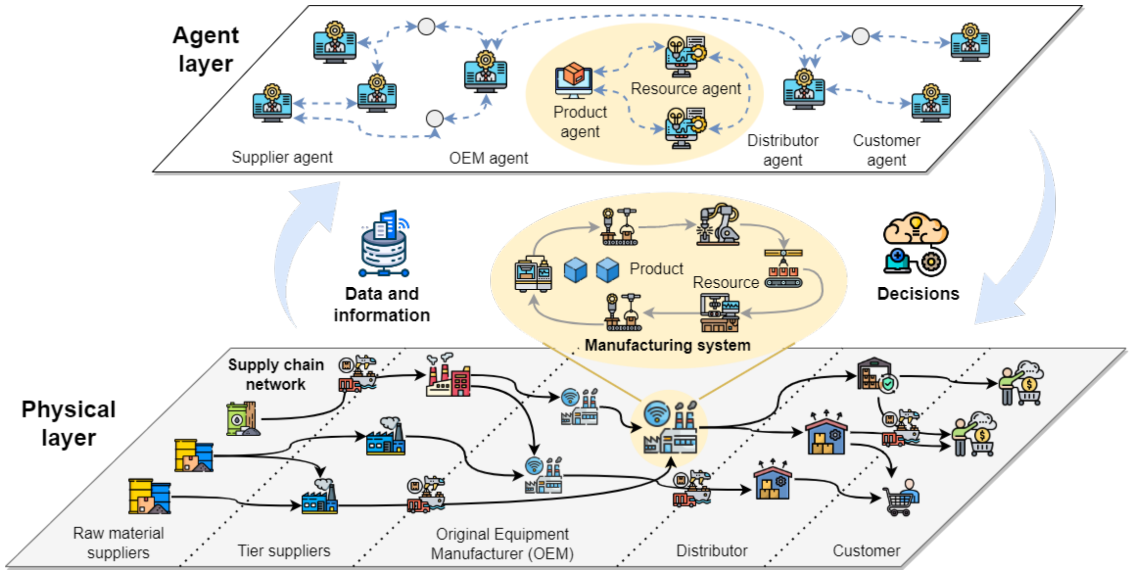

Figure 1.1: An overview of a general multi-agent framework in manufacturing systems and supply chain networks.

communication and collaboration for system-level decision-making, have been proposed to improve the flexibility and agility of industrial systems [44–47].

Multi-agent system theory has been studied to enable intelligent and distributed decision-making in manufacturing systems [3, 30] and supply chain networks [10, 32]. Derived from the disciplines of Distributed Artificial Intelligence (DAI) and computer science, intelligent software agents are defined as autonomous computational entities that represent physical or logical objects in the system, pursue their own goals, react to their environments, and communicate with other agents [48]. A multi-agent system consists of various autonomous agents that communicate and interact with each other to make high-level decisions for the behaviors of their associated physical objects based on their knowledge and goals [49]. These high-level decisions of the agents drive the performance of the physical layer [3, 32]. Depending on different industrial environments, industrial agents could represent different system entities, such as the products and machines in manufacturing systems or suppliers and customers in supply chains, as shown in Figure 1.1. Emerging techniques, such as cloud computing and RFID, enable agents to collect data from the physical layers continuously, communicate with each other, and convey decisions on the fly [19, 34, 43].

Although researchers have developed numerous multi-agent approaches for industrial systems, most approaches either focus on initial scheduling and planning [50–53] or assume prior knowledge of potential disruptions and cannot react to unexpected disruptions [37, 38]. In addition, the design of the agents is mostly based on conceptual models [54, 55] or rule-based decision-making [56–58]. However, these architectures are difficult to update dynamically as the environment changes and scales as the system becomes larger and more complex [6, 44]. Therefore, to apply multi-agent approaches to large, complex, uncertain, and dynamic industrial systems in the presence of multiple different disruptions (e.g., machine failures, material shortage, demand shifting, etc.), some challenges and gaps must be addressed: 1) developing a general model-based agent architecture that is flexible and scalable for different physical entities; 2) designing a communication strategy that enables agents to collaborate in an agile and efficient manner in the presence of multiple different disruptions; and 3) creating models of agent decision-making with uncertainty and risk awareness to achieve system-level resiliency.

1.2 Dissertation Overview and Contributions

The objective of this dissertation is to **improve the agility and resiliency of industrial systems in the presence of various unexpected disruptions through a distributed decision-making strategy**. To cope with the challenges and gaps listed above, this dissertation proposes a general multi-agent framework and applies it to two different levels of industrial environments: manufacturing systems and supply chain networks, aiming to address different disruptions ranging from small-scale resource breakdown to large-scale product market shifts. The proposed multi-agent framework comprises various model-based agents, dynamic agent communication strategies, and optimization-based decision-making.

The proposed model-based agent architecture allows agents to update their knowledge about themselves and their local environments dynamically. When agents need to make decisions, they retrieve information from their current knowledge base and exchange information through the

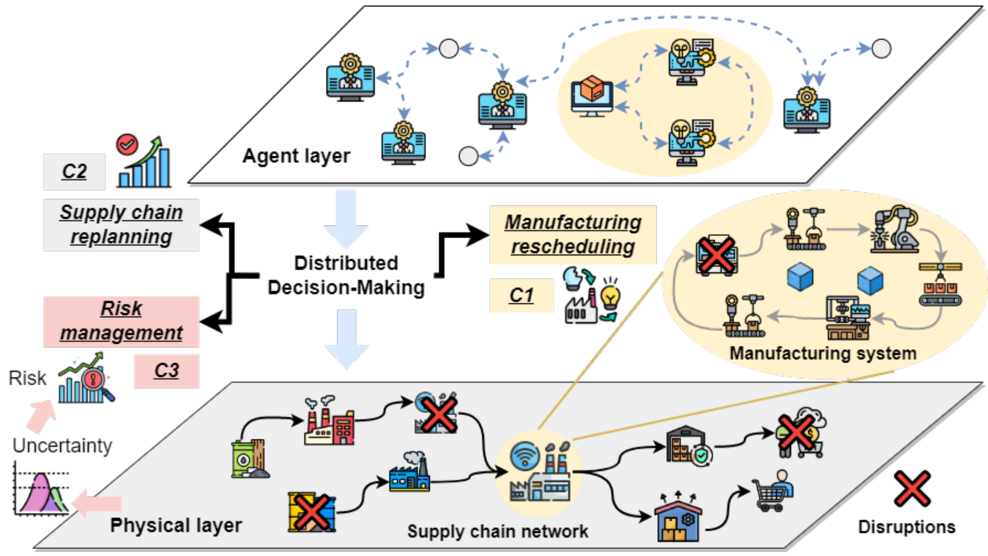

Figure 1.2: An overview of the contributions of this dissertation: C1) a dynamic and resilient rescheduling strategy for manufacturing systems; C2) a heuristic-guided dynamic re-planning strategy for supply chain networks; and C3) a heterogeneous risk management mechanism for resilient disruption response for supply chain networks.

designed communication strategy. Based on their knowledge and communication information, the agents can dynamically identify new actions to respond to unexpected disruptions. The system performance can be recovered agilely without prior knowledge or predetermined behaviors. In addition, the developed decision-making algorithms incorporate the uncertainties and risks of the agent and its surrounding environment to improve the resiliency of the systems.

The core contributions of this dissertation can be described by how the proposed framework is applied to different industrial systems (i.e., manufacturing systems and supply chain networks), as shown in Fig. 1.2. In this dissertation, we start with manufacturing systems since it is directly related to the production activity and within a relatively constrained environment. Then we extend to supply chain networks that are larger, more complex, more dynamic, and more heterogeneous than manufacturing systems. The specific contributions are described as follows:

C1: A dynamic and resilient rescheduling strategy for manufacturing systems:

Most existing multi-agent frameworks for manufacturing systems include product agents (PAs) and resource agents (RAs) [3, 45, 50]. A PA is responsible for fulfilling production requirements for its associated physical product, while an RA provides high-level control for its associated resource in the physical layer [50]. For the rescheduling problem, existing approaches use either PA-RA coordination [51, 59–61] or RA-RA coordination [62–65]. However, these approaches either cause large deviations between new and initial schedules or have limited agility due to unnecessary coordination. In addition, these prior studies build agents in a static and deterministic manner, which cannot support dynamic on-the-fly decision-making and the incorporation of uncertainties and risks for the rescheduling problem.

Therefore, the **first contribution (C1)** of this dissertation is a dynamic and resilient rescheduling strategy via capability-based clustering communication and risk assessment for internal disruptions in manufacturing systems. A simulated manufacturing system with 20 machines and 6 mobile robots to produce 100 products is implemented. Considering the machines could break down at any time, the proposed strategy realizes an **11%** improvement in throughput recovery and reduces around **50%** total breakdowns compared to the strategy without risk assessment. The proposed multi-agent manufacturing system, rescheduling strategy, and case study validation are presented in Chapter 2 and in [11, 66].

However, disruptions in industrial environments are often larger and more complex than resource disruptions in manufacturing systems. The recovery from these disruptions, such as material supply shortages, customer demand shifts, and production line shutdowns, might be limited or even infeasible if we only focus on individual manufacturing systems. Such disruptions require entities from the supply chain network to join the response decision-making process. We apply the proposed multi-agent framework to supply chain networks in response to such disruptions.

C2: A heuristic-guided dynamic re-planning strategy for supply chain networks:

Compared to manufacturing systems, supply chain networks are usually highly heterogeneous, larger-scale, less-constrained environments. Most existing agent-based supply chain disruption response methods are based on prior knowledge of disruptions. The disruption scenarios and response actions are pre-defined via a stochastic programming model [67] or a Petri Nets model [68]. Researchers use past information to generate a case-based disruption response database [37]. Therefore, the disruption responses of these methods are limited to the set of pre-defined scenarios or historical cases, and it is difficult and even impossible to cope with unexpected disruptions. In addition, existing agents are equipped with rule-based reasoning [57, 69, 70] or case-based decision-making [29, 58, 69], which limits the agility in response to supply chain disruptions. Therefore, the **second contribution (C2)** of this dissertation is a heuristic-guided dynamic re-planning strategy that enables agent exploration and iterative communication to respond to various supply chain disruptions. The performance is tested through a case study of an automotive cockpit supply chain with 6 different models, 3 different plants, and 84 supplier agents at different tiers. Considering a failure of any one of the 84 agents, in 42 cases, the distributed solution arrived at the same (or similar) solution as the centralized (optimal) one, with only 15% of the required communication and computation. The proposed multi-agent supply chain network, re-planning strategy, and case study validation are presented in Chapter 3 and in [17].

Though equipped with the ability to respond to a disruption, the proposed multi-agent frameworks are designed based on deterministic characteristics. However, agent knowledge and communication information, such as production cost and delivery time, are stochastic in practice. These uncertainties may result in negative effects (i.e., risks) on the industrial environments and should be considered to improve the system resiliency.

C3: A heterogeneous risk management mechanism for resilient disruption response for supply chain networks:

The performance of the disruption response obtained from deterministic decision-making cannot be guaranteed in real-world stochastic environments. In the supply chain domain, most research focuses on centralized risk management methods [39, 71, 72], which are difficult to scale up for handling supply-chain issues at the national or global level. In existing multi-agent methods, uncertainties and risks are considered to improve supply chain design [73] and address inaccurately predicted demand [74], but have rarely been considered in the decision-making for rescheduling and disruption recovery methods [7]. In addition, there is a high heterogeneity among different agents regarding their sensitivity to different risks and overall risk attitudes, which could change dynamically. However, current approaches either build a holistic risk model for the entire system [39, 71, 72] or assign identical risk attitudes for all agents [75–78]. These methods do not allow agents to have different and/or dynamic risk models depending on the information they obtain. Therefore, the **final contribution (C3)** of this dissertation is a heterogeneous risk management mechanism for agent decision-making that improves the resiliency of the network to unexpected disruptions. The proposed heterogeneous risk management mechanism is presented in Chapter 4.

In summary, the proposed model-based agent architecture lays a foundation for employing dynamic decision-making. The developed heuristic-guided agent communication strategies improve the agility of manufacturing systems and supply chain networks. The incorporation of risk management enhances their resiliency, especially in the presence of unexpected disruptions. From an industrial perspective, disruptions occur more frequently and unexpectedly as industrial environments become more dynamic and complex. For small-scale disruptions that either only impact a part of the system or do not last a long time, the proposed approach can compute a good response quickly through local negotiations. For large-scale disruptions, it may be worthwhile to re-optimize the entire system using centralized approaches, but enterprises can still apply this work to conduct a quick temporary recovery plan to minimize the negative effects, such as time effort and monetary loss. In addition, this work could be integrated into a Decision Support System (DSS) to test different scenarios that would be too complex and risky to be tested in the real world. This information could provide insights and flexibility for practitioners to investigate the system performance with various systematic parameters, such as disruption types, system attributes, and objectives.

From the research perspective, this dissertation provides a generalized way to design a multi-agent system to realize distributed intelligence, including agent architecture, coordination strategies and protocols, and decision-making capabilities. In addition to manufacturing systems and supply chain networks, the proposed framework is transferable to other complex systems, such as multi-robot systems, autonomous vehicles, and human-robot teams, that consist of multiple intelligent entities.

The rest of the dissertation is structured as follows. Chapter 2 provides the multi-agent framework for dynamic and resilient rescheduling in manufacturing systems and performance evaluation through a simulation case study. Chapter 3 presents the multi-agent framework for the heuristic-guided dynamic re-planning in supply chain networks and performance evaluation through case studies. Chapter 4 describes the heterogeneous risk management mechanism for resilient disruption response for supply chain networks. Chapter 5 states the conclusions and broader impacts of this dissertation.

CHAPTER 2

Multi-Agent Framework for Rescheduling in Manufacturing Systems

This chapter presents an architecture for model-based resource agents and a dynamic and resilient rescheduling strategy in a manufacturing system. The results have been published in [11, 66]. As described in Chapter 1, a number of multi-agent architectures have been proposed for the control of manufacturing systems. Most of the existing literature on multi-agent control for manufacturing systems focuses on rule-based agent communication behavior and deterministic manufacturing environment for the rescheduling problem, which limits the flexibility, agility, and resiliency of the decision-making process. In this chapter, we focus on a rescheduling problem in the presence of a resource breakdown in manufacturing systems. To improve the performance requirements, we introduce an RA architecture that enables a capabilities-based clustering scheme and a risk assessment approach for dynamic and resilient resource reallocation.

The rest of the chapter is organized as follows. A literature review and problem formulation are presented in Section 2.1. Section 2.2 describes the proposed architecture of RAs. The agent communication and decision-making for rescheduling are discussed in Section 2.3. Section 2.4 presents a simulation case study with the proposed RA architecture and rescheduling strategy. Concluding remarks are presented in Section 2.5.

2.1 Background and Overview

In this section, a literature review of the multi-agent approach for the rescheduling problem in manufacturing systems is provided. Then an overview of the proposed formulation of the rescheduling problem is presented.

2.1.1 Literature Review

The scheduling and rescheduling problems have been widely studied via centralized decision-making, such as mathematical programming [79, 80] and reinforcement learning [24, 25, 27, 81, 82]. However, centralized decision-making with all the factory information might be inefficient in quickly responding to the dynamic manufacturing environments. Therefore, multi-agent architectures with distributed decision-making have been introduced in manufacturing systems to improve flexibility and agility when solving the scheduling and rescheduling problem [3, 44, 45, 50, 59].

Most of the existing multi-agent architectures identify the roles of various manufacturing system agents and develop their communication and decision-making requirements for the control strategy. Product agents (PAs) and resource agents (RAs) have been described in most existing multi-agent architectures [51]. A PA is responsible for fulfilling production requirements for its associated physical part through interactions with other agents, while an RA provides high-level control for its associated resource in the physical layer [50]. Through the coordination and decision-making of PAs and RAs, flexibility and responsiveness in manufacturing systems can be improved [3]. Figure 2.1 shows an overview of a general multi-agent manufacturing system that includes product agents, resource agents, and the factory floor.

Some existing multi-agent architectures consist of agents who are responsible for making scheduling decisions after collecting information from PAs and RAs. The contact agent introduced by [59] and the rescheduling agent developed by [83] receive resource disruption information from the disrupted RA and then start the rescheduling process with knowledge of the entire system. However, these types of agents essentially provide centralized decision-making for scheduling,

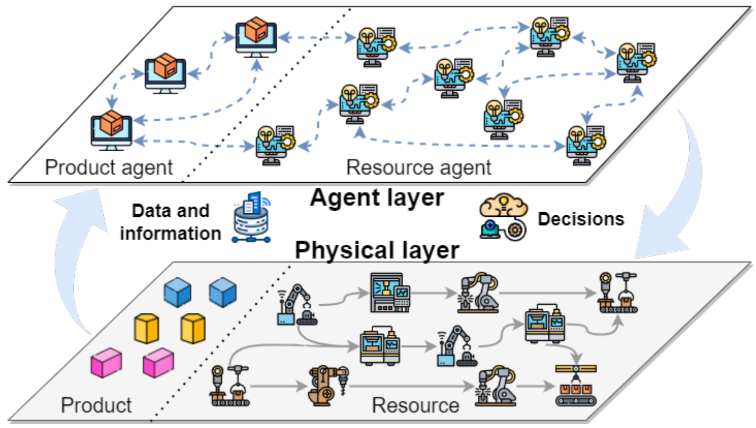

Figure 2.1: An example of a multi-agent manufacturing system that comprises product agents (PAs) and resource agents (RAs).

which has limitations in quickly responding to dynamic environments. Therefore, this chapter focuses on the distributed decision-making process via agent coordination to solve a rescheduling problem and quickly compute a resilient new schedule.

In existing multi-agent architectures, it is commonly stated that RAs are the class of agents that identify resource disruptions by continuously collecting data from their associated resources, while the decision-making process for rescheduling could be triggered by a PA or an RA. In the studies [51, 60, 61], PA triggers the rescheduling process and determines a new schedule. Once the disrupted RA informs the PA of a need for rescheduling, the PA sends a rescheduling request and triggers the PA-RA coordination to generate a new resource allocation schedule based on the remaining tasks and resource capacities. In [51], the RAs receiving the PA's request will propagate the request to other RAs if they cannot satisfy the requirements. However, these methods do not try to preserve the initial schedule; thus they have a high probability of causing deviations between the new and initial schedule. In the rescheduling problem, the deviation between the new and initial schedule is defined as scheduling robustness [84]. Thus, the methods from [51, 59, 60] have limited scheduling robustness.

To address the limitation, determining modifications to the initial schedule should be considered. Therefore, for resource disruptions, RA coordination can be applied by using the local view of the RAs. In [85], researchers introduce a reconfiguration agent as a mediator for RA coordination to respond to the reconfiguration and communication requests from different RAs. For direct RA coordination, some collaborative mechanisms are provided to enable a disrupted RA to request all of the other RAs [62] or all RAs of the same type [63] to make reallocation decisions. However, it is not necessary for the disrupted RA to communicate with all RAs since some RAs do not have the required capabilities to perform the affected operations of the disrupted RA. Therefore, these methods create a significant communication load that will limit the agility of the system in response to a disruption. To reduce agent communication, clustering approaches have been used to provide a structured coordination process. In [64], an RA cluster is defined as a set of RAs that collaborate to complete a sub-task. In [65], researchers define an RA cluster based on both physical constraints and resource proximity. However, for a rescheduling problem, fixed coordination rules and considering only nearby resources might cause resource overload or no alternative resource to be found, which reduces throughput and resource utilization.

To cope with the problem, the disrupted RA needs to dynamically determine the agents it coordinates with (i.e., RA cluster) instead of following a pre-defined rule-based coordination strategy since the rescheduling scenarios are highly variable. The environment information, such as other agent attributes and coordination behaviors, should be designed and included in the RAs' knowledge base. The existing studies [59, 63, 85, 86] focus on how an RA makes decisions to respond to other agents' requests through their proposed modularized RA architectures. In [87], a reinforcement learning approach is introduced to enable agents to learn the environment to solve the scheduling problem. However, these methods do not cover how the agents can dynamically determine their coordination behaviors for the rescheduling problem.

In addition, it is important to consider the rescheduling resiliency or robustness, which refers to the ability to recover system performance under different environmental conditions such as uncertainties [88]. In many systems, the quality and execution of a rescheduling solution lie in the resiliency and robustness of the new schedule. Thus, to ensure a dynamic and resilient response, the decision-making process must incorporate uncertainty and potential risk factors into the optimization. However, most scheduling/rescheduling methods with risk assessment focus on robust scheduling, which refers to deriving schedules that are resilient to disruptions [84]. In [88–90], risk scenarios are incorporated into the Petri Net model or automata of the entire system and are considered when the system generates a production schedule. In [23], researchers provide an algorithm for robust scheduling considering uncertain processing times. In [91], a conceptual structure that enables risk assessment in production scheduling is introduced. These studies primarily focus on risk assessment in the process of generating an initial schedule and obtaining a resilient schedule in the presence of disruptions. However, some disruptions will require the development of a new schedule to meet the process throughput. Therefore, incorporating risk assessment into the rescheduling process is an important need. However, the current studies use centralized methods to cope with risks for the rescheduling problem [92, 93], while none of the existing distributed rescheduling methods incorporate risks in their decision-making process.

In summary, for the rescheduling problem, existing multi-agent decision-making methods do not currently satisfy the following needs defined to achieve agile and resilient rescheduling: (1) minimization of changes to the original production schedule, (2) dynamic and distributed decision-making via agent coordination, and (3) incorporation of metrics that quantify risks into distributed rescheduling decision-making.

To address these limitations, the contributions of this section include: (1) the extension and generalization of an RA architecture that includes a Knowledge Base, a Communication Manager, and a Decision Manager, (2) the development of a capabilities-based clustering scheme and a risk assessment approach for dynamic and resilient resource reallocation, and (3) an evaluation of manufacturing system performance when implementing the proposed approach within a simulated manufacturing facility.

2.1.2 Problem Overview and Formulation

This section provides formal definitions of the multi-agent architecture and components within a production schedule. A resource reallocation problem in the form of a rescheduling task is also formulated.

Table 2.1: Nomenclature for the RA architecture

Production schedule	
$\mathcal{S}$	Function that maps agents to the product and resource schedule
s	Function that maps each agent to a sequence of events in the schedule of the agent
Ag	Function that maps events to particular agents
T_s	Function that maps event sequence to the start and end times of each event
Agents	
X	Set of states of a product
E	Set of events representing operations that change product state
T_r	State transition function representing how events drive state changes
$x_i = (x_i^\ell, x_i^c)$	State of a product that describes its location and physical composition
T	Cost function for performing events
A_t	Function that maps events to resource attributes
P_q	Set of production requirements for scheduled events
C_ℓ	Function that maps events to clustering RAs
Rescheduling process	
RA_d	The disrupted RA
E_d	Sequence of affected events in the resource schedule of RA_d
s_d	Sequence of events that need to be replaced
x_{prior}	State before the event sequence that should be replaced (s_d) in the initial product schedule
x_{post}	State after the event sequence that should be replaced (s_d) in the initial product schedule
s_{new}	Sequence of events that can replace the event sequence s_d
H	Function that calculates the earliest available time for a resource to perform an event
R	Function that calculates the risk of a new event sequence

2.1.2.1 Definitions

Manufacturing system – resources that are connected by material and information flow with a control architecture to produce finished goods [44].

Resources – the entities, such as humans or equipment, that perform operations (e.g., production, maintenance, and transportation) in a manufacturing system.

Central knowledge base – contains all the information relevant to the manufacturing system, such as product requirements, resource capabilities, etc. It is initialized by the manufacturer.

Production goal – an objective to transform raw materials into finished products to meet customer demands through certain resource operations.

Production schedule – a plan that specifies resources to perform operations on parts at certain times to achieve the production goal. A detailed definition is stated in the following sections.

2.1.2.2 Agent formulation

In this dissertation, PAs and RAs are used to describe the multi-agent manufacturing system and outline the rescheduling problem.

Product agent A PA is responsible for fulfilling the desired production requirements of its associated physical product. The study in [50] introduces a model-based PA architecture that enables PAs to make intelligent decisions to guide products and track the production progression through the manufacturing system. Each PA stores the status of its associated product as a discrete state in the set $X = \{x_0, x_1, ..., x_f\}$, where x_0 is the initial state and x_f is the final state of the product in the manufacturing system. Each state is comprised of two elements, $x_i = (x_i^\ell, x_i^c)$, where x_i^ℓ and x_i^c denote the product's location and physical composition, respectively. Note that precedence constraints may exist in the physical composition states x_i^c while usually not in the location states x_i^ℓ. For instance, a PA state can be represented as $x_i =$ ("at machine1", "with a milled pocket").

Resource agent An RA provides high-level control for a physical resource to perform operations on products. In this work, RAs are grouped into two RA classes: transportation and transformation RAs, based on the operations they can perform on the products. The resource operations are modeled as a set of discrete events, denoted by $E = \{e_0, e_1, ..., e_n\}$. An event for a transportation RA results in a state change in the location of a product, while an event for a transformation RA results in a change in the physical composition. More information regarding the resource agent is discussed in Section 2.2.

2.1.2.3 Production schedule

From the definitions above, the production schedule for the manufacturing system is a collection of schedules for all p products (or equivalently, all r resources) in the manufacturing system. The production schedule for a PA or an RA contains different information. The set of all PAs and RAs in the system is denoted by $\mathcal{A} = \{PA_0, ..., PA_p, RA_0, ..., RA_r\}$, where p is the number of PAs and r is the number of RAs in the system. The production schedule for each agent is calculated by a function:

$\mathcal{S} : \mathcal{A} \rightarrow (s, Ag, T_s)$, where

$s : \mathcal{A} \rightarrow e_0, ..., e_a$: is a function that maps agents to the sequence of events scheduled to be performed either on that product or by that resource

$Ag : s \times \mathcal{A} \rightarrow \mathcal{A}$: is a function that represents the relationship that describes the events, the RAs that perform the events, and the PAs on which the events are performed

$T_s : s(\mathcal{A}) \rightarrow (\mathbb{R}_+, \mathbb{R}_+)$: is a function that maps events to start and end times

For a given resource, the event sequence $s(RA_j)$ represents the events that RA_j will perform, and Ag provides the PA on which the events are performed. $T_s(s(RA_j))$ provides the start and end times for each event in the resource schedule. It is assumed that a resource cannot perform multiple events at the same time, thus, there should be no intersections (and generally should exist time gaps) between event time periods for a given resource. Therefore, one might define an idle time interval between the end time of one event and the start time of the next event. The set of idle time intervals, denoted by $I = \{[t_0, t_1], [t_2, t_3], ...\}$, can be calculated for each (re)scheduling purpose. The production goal will be achieved if the specified resources follow their designated schedules for each product in the system.

For a specific product, the event sequence $s(PA_i)$ defines the PA state transitions from the initial state, x_0, to the final state, x_f. To represent how an event represents a change in the state of a PA_i, a state transition function is defined as $Tr : X \times E \to X$. The PA states and event sequence, $s(PA_i)$, satisfy the transition relationship:

$$x_f = T_r(x_0, s(PA_i)). \tag{2.1}$$

Based on the transition relationship, the start and end times in $T_s(s(PA_i))$ indicate the time periods during which the product is associated with a specific state. Since the product state is always defined, the times provided in $T_s(s(PA_i))$ will not contain any time gaps. Hence, the end time of one event (or state) equals the start time of the next event (or state). Note that an event type (e.g., milling a pocket) can occur multiple times in $s(PA_i)$, but at varying occurrence times and with different RAs. The function Ag identifies the specific RA that is associated with a particular event being applied to PA_i. Note that every event in the schedule of a given product is also an event for the associated resource and vice versa. However, the indices of the specific event are not the same within the product and resource schedules.

2.1.2.4 Problem statement

Resource allocation can be formulated as a production scheduling problem [45]. When unexpected resource disruptions (e.g., breakdowns) occur in dynamic manufacturing systems, the initial production schedule cannot be executed as originally planned [38]. Therefore, the products that are impacted by this disruption may be rescheduled through the reallocation of resources [21].

This rescheduling problem is outlined as: given a manufacturing system (r resources) with a production goal (p products to produce) and feasible initial production schedule ($\mathcal{S}$), assuming

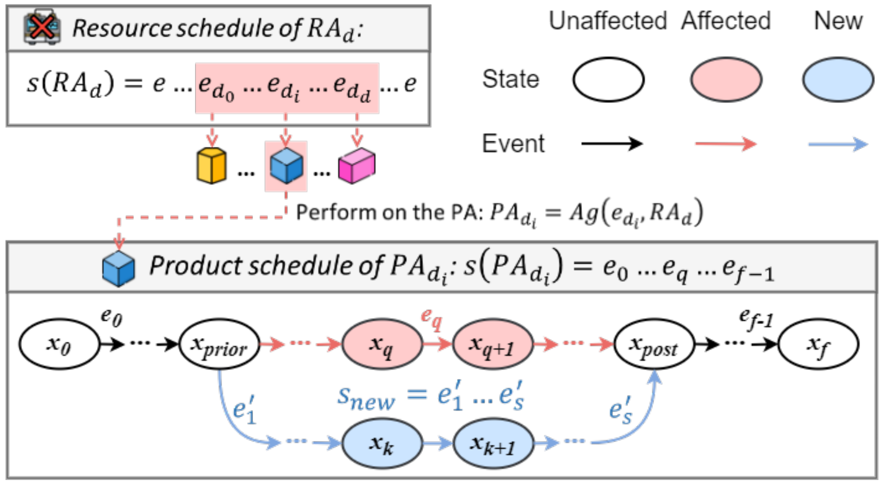

Figure 2.2: Discrete event system representation of the problem formulation. Each affected event e_{d_i} from the affected resource is associated with a specific PA (PA_{d_i}), where the PA schedule denotes the index as q. e_q cannot be performed on the product, and thus the state transition for PA_{d_i} from x_{prior} to x_{post} cannot be achieved. The reallocation problem is to find a new event sequence s_{new} that can recover this transition.

a single resource breaks down (RA_d), find a new feasible and resilient production schedule on-the-fly that minimizes changes to the initial schedule S and optimizes user-defined objectives. To formulate the problem, the following assumptions are provided:

A.1 The initial production schedule is predetermined and will achieve the production goal if followed.

A.2 Unexpected resource disruptions are detectable by the associated RAs and result in the specific resources becoming unavailable for a certain amount of time.

A.3 The manufacturing system contains resource redundancy and is operating with available capacity.

A.4 The rescheduling time can be neglected compared to operation time.

A.1 ensures that the manufacturing goal can be met if the rescheduling process follows the production requirements in the initial schedule. A.2 guarantees that a disruption will be identified by a resource if it occurs and also designates how a resource will be impacted by the disruption.

Algorithm 1 Identify the shortest event sequence that needs to be replaced

Input: e_{d_i}, RA_d, s, Ag
Output: s_d
// Identify the index of e_{d_i} in its associated product schedule
$PA_{d_i} \leftarrow Ag(e_{d_i}, RA_d)$
$j \leftarrow 0$ and $e_j \in s(PA_k)$
while $e_j \neq e_{d_i}$ or $Ag(e_j, PA_k) \neq RA_d$ **do**
    $j \leftarrow j + 1$
    $q \leftarrow j$
end while
// Find the event sequence that needs to be replaced
Add e_q to s_d
// For the events e_j before e_q in $s(PA_k)$
$j \leftarrow q - 1$
while $x^{\ell}_{j+1} = x^{\ell}_d$ and $j \geq 0$ **do**
    Add e_j to the first position of s_d
    $j \leftarrow j - 1$
end while
// For the events e_j after e_q in $s(PA_k)$
$j \leftarrow q + 1$
while $x^{\ell}_{j+1} = x^{\ell}_d$ and $j \leq f - 1$ **do**
    Add e_j to the last position of s_d
    $j \leftarrow j + 1$
end while
return s_d

A.3 is necessary to enable agent coordination and part rerouting. A.4 simplifies the rescheduling problem by assuming there are no changes in the manufacturing system during the decision-making process.

Once a resource disruption occurs, the associated RA is able to identify the disruption (A.2) and determine the events that the resource can no longer perform, denoted by E_d, which is a sub-sequence[1] of the original event sequence for resource RA_d: $E_d \subseteq_{seq} s(RA_d)$. All of the events in E_d need to be re-assigned to alternative resources, which requires resource redundancy and available capacity (A.3).

As shown in Fig. 2.2, each event $e_{d_i} \in E_d$ belongs to the schedule of its associated PA, denoted by $PA_{d_i} = Ag(e_{d_i}, RA_d)$. Since RA_d cannot perform e_{d_i}, PA_{d_i} cannot achieve its production goal

[1]For simplicity, the symbol $\subseteq_{seq}$ is used to represent the sub-sequence relationship in this dissertation.

(i.e., state transitions in Eqn. (2.1)). The sequential events associated with e_{d_i} in a given product schedule may become unnecessary (e.g., transportation events to/from the broken machine). We define s_d as the shortest sequence that contains e_{d_i} and should be replaced by a new event sequence s_{new} in the production schedule. To identify the sequence s_d, the index of e_{d_i} for the specific product, PA_{d_i}, is denoted by q (i.e. $e_{d_i} = e_q$). Algorithm 1 determines s_d by checking whether the associated states of the sequential events are related to RA_d. In this way, s_d is guaranteed as the shortest sequence that contains e_{d_i} and needs to be replaced. Once s_d is identified, two states x_{prior} and x_{post} are defined as the states before and after s_d in the product schedule of PA_{d_i}, where $Tr(x_{prior}, s_d) = x_{post}$.

Therefore, for each affected event e_{d_i} and its associated product PA_{d_i}, the rescheduling process aims to search for a new sub-sequence of events (s_{new}) that includes the events that need to be replaced, s_d:

$$Tr(x_{prior}, s_{new}) = x_{post} \tag{2.2}$$

Note that the affected event e_{d_i} being performed by an alternative RA should be an element of s_{new}, and the new sequence should satisfy the production requirements.

Through Algorithm 1 and Eqn. (2.2), the rescheduling problem is formulated in a way that minimizes the changes to the initial schedule. Therefore, instead of resolving a system model to generate a fully new optimal schedule (e.g., job shop schedule, which is NP-hard), we focus on modifying the initial schedule by locally searching for an alternative resource to replace the broken resource to recover the performance and thus minimize the impact to the initial schedule. It requires less computational effort than re-generating the total schedule while it loses some optimality. In this case, the problem in this work is polynomial-time solvable since the worst case is to evaluate all the resources in the system for each event that needs to be replaced. This process takes $O(r \times \sum_{e_{d_i} \in E_d} |s_{d_i}|)$ computations, where r is the number of RAs in the system. A centralized method can be applied to provide an optimal schedule based on the performance objective (e.g., throughput); however, this approach requires significant computation efforts to achieve centralized optimization, making it less agile in many disruption scenarios. In this work, we propose an RA

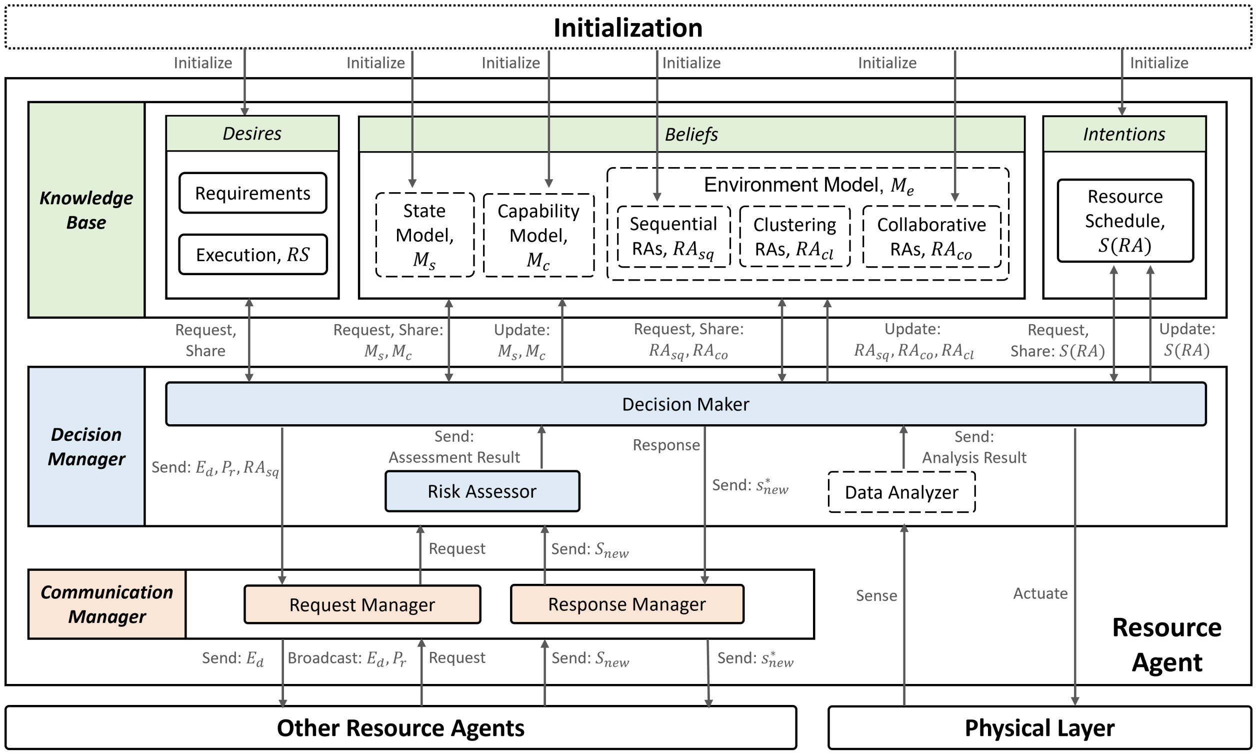

Figure 2.3: The internal resource agent architecture, including Knowledge Base, Communication Manager, and Decision Manager. The communication between each component for the rescheduling problem is also displayed.

communication strategy with capability heuristics to avoid the need to communicate and optimize across all of the resources, thus reducing computational efforts. We then incorporate risk assessment into the rescheduling decision-making problem to investigate how the consideration of risk improves throughput.

2.2 Resource Agent Architecture

2.2.1 Knowledge Base

The Belief-Desire-Intention (BDI) architecture has been widely used to provide a modular framework to design intelligent agents [94]. Following the BDI design, the model of an RA in the authors' previous work is contained in the beliefs segment of the architecture within this work. In this work, we reformulate the structure and content of the belief section of the RAs as the desires and intentions are developed and integrated into the Knowledge Base. As shown in Fig. 2.3, several aspects of the Knowledge Base are initialized before the manufacturing system begins operating. We assume this initialization is completed by the manufacturers based on the customer order, physical layer, and initial production schedule.

2.2.1.1 Intentions and Desires

Intentions Agent intentions represent the plan an agent has committed to execute. In this section, the intentions of an RA are represented by the resource schedule $\mathcal{S}(RA_j) = (s, Ag, T_s)$, as defined in Section 2.1.2.

Desires Agent desires represent the goal and requirements of an agent. As shown in Fig. 2.3, the desire of an RA is to execute the resource schedule $\mathcal{S}(RA_j)$ without violating requirements for production and safety.

In this section, function $P_q : E \times PA \rightarrow Requirements$ maps each scheduled event for a given product agent to its specific production requirements (e.g., precision). The production requirements that RA_j must satisfy based on the products that will engage with the given RA are represented as a set $\{P_q(e_i, Ag(e_i, RA_j)) : e_i \in s(RA_j)\}$. These requirements are then split into hard and soft requirements. Hard requirements must be followed while soft requirements can be negotiated to meet the demand with the introduction of a penalty. For example, a hard requirement might be the size constraint of a product that can be assigned to a resource such that the product will fit within the workspace of the resource. A soft requirement could include the bound on the energy cost for a given event that may need to be violated in order to meet the product due date [95].

The intentions and desires are related to the resource schedule, thus they are assigned to RA_j once the initial production schedule of the manufacturing system is determined, and will be updated as the resource schedule changes.

2.2.1.2 Beliefs

Building from the architecture used in previous work [11], the beliefs of an agent are comprised of the state, capability, and environment models. These models are dynamically updated (i.e., extended, shrunk, and revised) as the resource and its environments change.

State model The agent state model describes how an RA monitors the status of the associated physical resource. Researchers in [96] introduce a Finite State Machine (FSM) framework to model the status of a manufacturing resource using several states and transitions. Similarly, the RA state model in this section is defined as an FSM that includes *Idle*, *Up*, and *Down* states as well as the transitions between these states, as shown in Fig. 2.4.

Transitions between RA states are triggered by the decision maker of the RA. As shown in Fig. 2.3, the sensor data in the physical layer is collected by the data analyzer, which utilizes this data to identify the current status of the physical resource. Though this work does not focus on data-driven analysis, related work has been done to achieve state and anomaly identification [97]. Having obtained the analysis results, the decision maker checks the current state model and decides whether an update to the state model is needed (e.g., trigger the transition to *Down* if the resource is broken).

Capability model The agent capability model provides a detailed description of the operations that a resource can perform on parts. As defined in Section 2.1.2, RAs are grouped into two RA classes: transformation RAs and transportation RAs. The resource operations can be modeled as discrete events that drive state changes in the parts, as shown in Fig. 2.4. Therefore, an FSM can be used to model the capabilities of an RA [51, 98]:

$M_c = (X, E, T_r, T, A_t)$:

$X = \{x_0, ..., x_n\}$: a set of states that can be achieved on products utilizing the resource

$E = \{e_0, ..., e_m\}$: a set of events representing operations that change product states

$T_r : X \times E \rightarrow X$: a state transition function

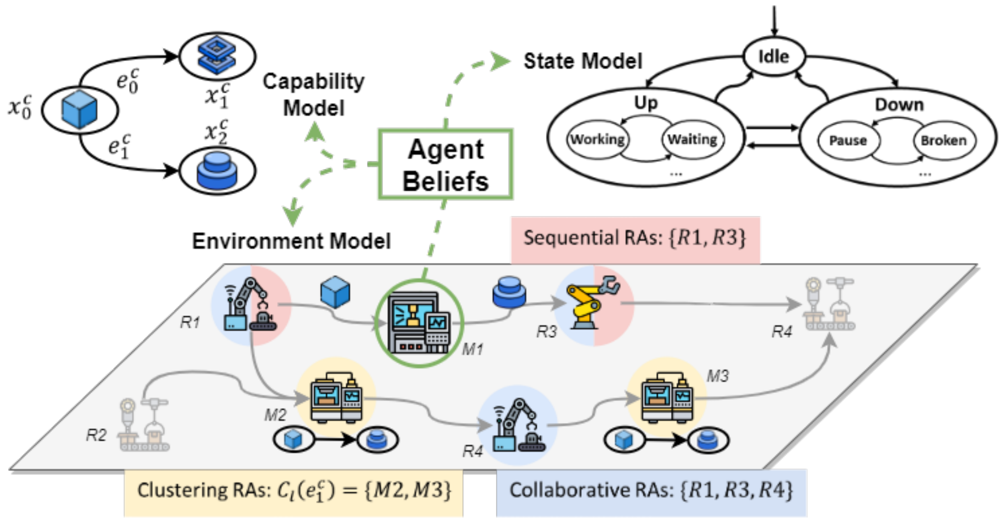

Figure 2.4: An example of the beliefs of a machine agent in a manufacturing system. The beliefs include the state, capability, and environment models.

$T : E \to \mathbb{R}_+$: amount of time associated with an event

$A_t(E, RA_j)$: a function that maps events and specific RAs to the physical resource attributes associated with each event (e.g., payload limitations)

In the capabilities model, the state set X contains all changeable states for the products associated with the given resource. E and T_r follow the definition provided in Section 2.1.2. The transition function T_r is inherited from the PAs. T represents the nominal cost (denoted as operation time) for each event to occur, assuming the cost for the same event is identical for different products. A_t provides the resource attributes for each event. Note that multiple RAs in the same class could have the same events, but the attributes might be different. The characteristics of the attributes are described by parameters, such as speed limitation, payload, and part dimensionality.

As shown in Fig. 2.3, the capability model is initialized based on the physical manufacturing system. Similar to the state model, as the data analyzer receives information constantly from the resource in the physical layer, the decision maker updates the capability model if there are any changes to the resource. The changes could be manual, such as tool replacement/removal, or spontaneous, such as machine breakdown.

Environment model The RA's knowledge of other RAs in the system is captured in the environment model. The relationships to these agents are modeled as mapping functions that map events or states to different sets of RAs, namely clustering, sequential, and collaborative RAs, as shown in Fig. 2.4.

Clustering RAs: The clustering RAs for RA_j are the set of RAs that can perform the same events as RA_j for a given subset of events E_s in RA_j's capability model. Each event in the given event subset $e_i \in E_s$ corresponds to a unique cluster. The relationship between each event e_i and the associated clustering RAs is modeled as a cluster mapping function, which is defined as $C_\ell : E_s \times RA \to 2^{RA}$, where

$$C_\ell(e_i, RA_j) = \{RA_k \mid e_i \in E_{RA_k},\ RA_k \neq RA_j\} \tag{2.3}$$

2^{RA} denotes the power sets of the RAs in the manufacturing system. E_{RA_k} represents the event set in the capability model of RA_k. Therefore, the set of $C_\ell(e_i, RA_j)$ maps represents the clustering RAs for the given event subset E_s for RA_j. As shown in Fig. 2.3, clustering RAs are not formed during the initialization. When clustering RAs are needed, RA_j retrieves the capabilities of other RAs from the centralized knowledge base and checks the constraints in Eqn. (2.3) to form the cluster map. Alternatively, RA_j can also request the capability information from the RAs within the manufacturing system to form the clusters dynamically.

Sequential RAs: The sequential RAs of RA_j depend on the resource schedule and associated product schedules. Every event in RA_j's schedule $e_i \in s(RA_j)$ corresponds to a specific product agent $PA_k = Ag(e_i, RA_j)$. In the product schedule of PA_k, the RAs that perform the events directly before and after e_i are sequential RAs of RA_j for this specific event. Each event $e_i \in s(RA_j)$ corresponds to a unique set of sequential RAs. To identify the sequential RAs, the index of event e_i in the product schedule of PA_k is denoted as q, which can be found following the same process in Algorithm 1. Note that q is bounded by $0 \leq q \leq f-1$ since the event sequence in the product schedule of PA_k is defined as $s(PA_k) = e_0 ... e_{f-1}$. Therefore, if e_i is the first or last event in the product schedule of PA_k, there is only one sequential RA for this event e_i. Otherwise, there are two sequential RAs. RA_j stores the information about the sequential RAs in a map that relates the scheduled event to specific RAs: $s(RA_j) \to 2^{RA}$. The set of the sequential RAs depending on the index q defined above:

$$
\begin{aligned}
&\{Ag(e_{q+1}, PA_k)\}, \text{ if } q = 0 \\
&\{Ag(e_{q-1}, PA_k)\}, \text{ if } q = f-1 \\
&\{Ag(e_{q\pm 1}, PA_k)\}, \text{ if } 0 < q < f-1
\end{aligned} \tag{2.4}
$$

The sets of $S_q(e_i)$ represent the sequential RAs for the scheduled events of RA_j. The set of sequential RAs is formed in RA_j's Knowledge Base based on the initial production schedule.As the system runs, the schedule of agents might need to change to adapt to disruptions, thus the sequential RAs should be updated as the schedule changes.

Collaborative RAs: To represent the collaboration between RAs, researchers in [51] introduce neighboring RAs, which have shared states in the capability model. In this work, collaborative RAs of RA_j are defined as the set of RAs that contain the same location states in their capability model. Each location state x_i^ℓ in RA_j's capability model corresponds to a unique set of collaborative RAs. This relationship is modeled as a mapping: $X \times RA \to 2^{RA}$. Therefore, for RA_j, its collaborative RAs in terms of x_i^ℓ is described as:

$$
\{RA_k \mid x_i^\ell \in X_{RA_k},\ RA_k \neq RA_j\} \tag{2.5}
$$

where X_{RA_k} is the state set in the capability model of RA_k. Note that a transportation RA can have both transportation and transformation collaborative RAs, while a transformation RA can only have transportation collaborative RAs. The set of collaborative RAs is formed in RA_j's Knowledge Base as the RA and its capability model are initialized, following the state relationship discussed above. This set of collaborative RAs is updated as the capability model changes. For example, if a mobile robot can no longer reach a machine at x^ℓ, the states related to x^ℓ will be removed from its capability model, as well as the collaborative RAs related to x^ℓ.

2.2.2 Decision Manager and Communication Manager

2.2.2.1 Decision Manager

The Decision Manager is responsible for the deliberation and reasoning process of an RA. Different decisions, such as product scheduling [50], RA response [86], etc., have been introduced in the literature. The Decision Manager in this work makes decisions about data analysis, scheduling management, communication, and risk assessment.

Data analyzer – a component that collects and analyzes data from the physical resource through sensors. The data analyzer may contain different data-driven models to abstract information that can be used by the agents from the raw data obtained from various sensors.

Risk assessor – a component that provides enhanced deliberation and reasoning processes to the RA by evaluating the risk of decision candidates. The risk assessor may contain different function models to analyze the risk of any decision based on the current status of the agent and the responses received from other agents.

Decision maker – a component that makes decisions regarding the execution of the current schedule and responds to requests from other agents based on the current status of the RA.

2.2.2.2 Communication Manager

The Communication Manager of an RA provides the interface between the RA and other agents for exchanging information. While the communication component has been mentioned in [50, 59, 63], these methods do not specify the different types of communication between the RAs. The Communication Manager in this work includes a request manager and a response manager.

Request Manager – a component that sends requests from the decision manager to other agents and passes requests received from other agents to the decision maker.

Response Manager – a component that sends the response from the decision manager to other agents and passes responses received from other agents to the decision maker.

2.3 RA Coordination and Decision-Making

In this section, the proposed rescheduling strategy via RA coordination is described. The coordination is guided by the agent environment models instead of following pre-defined rules. These models can be easily updated and scaled for different systems, thus the agent coordination behaviors are flexible and adaptable. An overview of the rescheduling process is shown in Fig. 2.5. In the agent coordination process, the constraints of the schedule are checked when agents determine their responses, and the new event sequence is augmented by propagating requests until the state transition is satisfied.

2.3.1 RA Coordination using Capability-based Cluster

2.3.1.1 Rescheduling request

When a resource breaks down, the associated RA, denoted by RA_d, must identify the breakdown, determine events that are affected by the breakdown, and create bid requests to start the rescheduling process.

Identify disruption and affected events A resource agent collects data continuously from the associated physical resource through sensors attached to this resource. The data is passed into the data analyzer within the Decision Manager of the RA. By feeding the data to the models in the data analyzer, the data analyzer identifies the current status of the physical resource and sends this information to the decision-maker. The decision maker then updates the knowledge base of the RA. When a resource is broken, the RA identifies the disruption and updates the state model to indicate a *Down* state.

After identifying the breakdown, the decision maker requests information about the resource schedule $S(RA_d)$ and production requirements P_q from the Knowledge Base. The decision maker will then determine the sequence of events, denoted as E_d, that need to be rescheduled. $E_d = e_{d_0}e_{d_1}...e_{d_d}$ is a priority event sequence, where each event corresponds to a priority value, which is calculated based on the original start time and the priority/importance of the associated product. An example priority mapping function could include a weighted sum of the inverse of the original start time and due date. For this example function, the order of the affected events in the sequence E_d will increase as the start times and/or due dates of a given product are extended. To realize dynamic rescheduling on the fly, the proposed method reschedules the affected events in a sequential manner following the order provided in E_d. For each affected event in E_d, RA_d runs Algorithm 1 to identify x_{prior} and x_{post}. For simplicity, the following description focuses on the rescheduling process for a single affected event e_{d_i}. This process will be repeated for each additional event within E_d.

Broadcast rescheduling request After identifying and sorting the affected events, the scheduling manager of RA_d sends a rescheduling request to the request manager. A broadcast technique is used for RAs to communicate information [49]. For each e_{d_i}, the request manager can dynamically identify the cluster RAs associated with e_{d_i} via the environment model and broadcasts the rescheduling bid request $Req = (e_{d_i}, P_q, x_{prior}, x_{post})$. P_q is the function that maps e_{d_i} to the production requirements. x_{prior} and x_{post} define the states that denote where a transition must be rebuilt, as defined in Eqn. (2.2).

As mentioned in Section 2.1.2, the sequential events of e_{d_i} may become unnecessary or transition to a different sequential event depending on the RA used to replace the affected event. In the case of a change to the sequential events, the request manager of RA_d sends the event e_{d_i} to the sequential RAs in $S_q(e_d)$ and requests them to remove the sequential events for e_{d_i} from their schedule.

2.3.1.2 Resource agent coordination

Cluster search Based on the resource capabilities, only the clustering RAs of RA_d with respect to e_{d_i} have access to the broadcast request. These RAs access the rescheduling request, and their

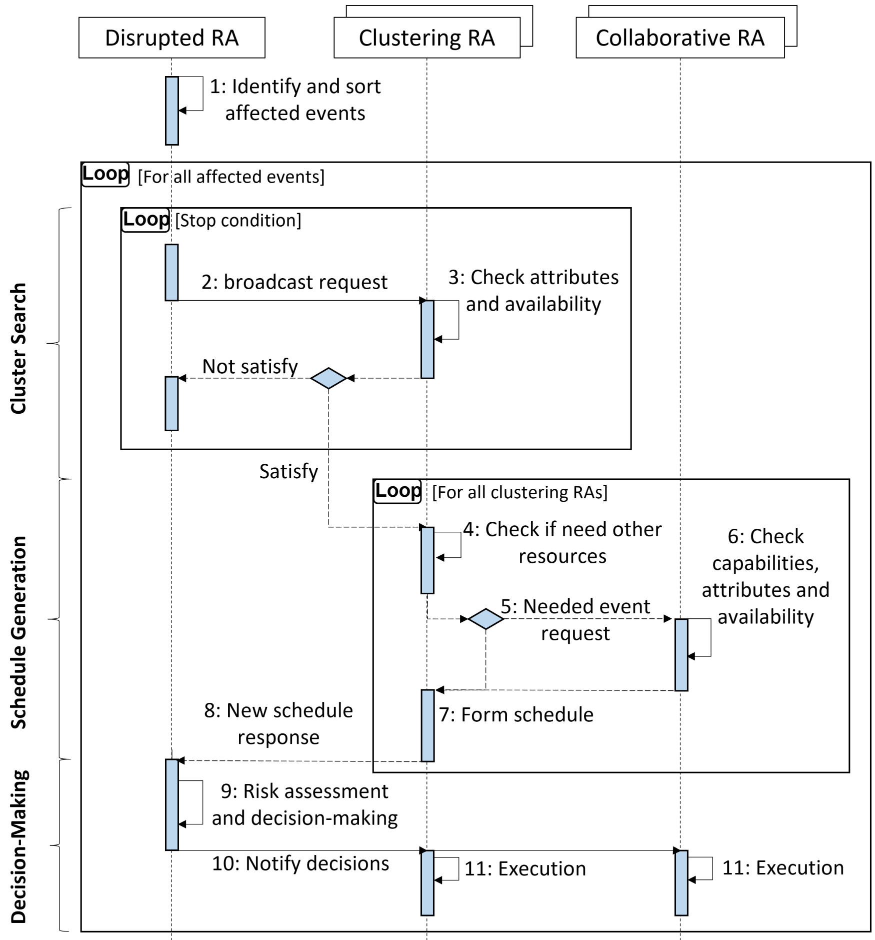

Figure 2.5: Coordination behaviors of resource agents for rescheduling process

request managers send the request to their decision maker. Through this clustering scheme, the agent coordination is more effective since RA_d requests the agents that can perform e_{d_i} instead of requesting all the other RAs or only nearby RAs. The decision maker requests the capability model from the Knowledge Base and conducts the following match-making steps:

- Check whether the RA still contains the affected event e_{d_i} in its capability model
- Determine whether the RA's associated resource attributes can satisfy the production requirements $P_q(e_{d_i})$

As defined in Eqn. (2.3), the first step ensures that a specific RA should be considered as a clustering RA of RA_d, while the second check determines whether the resource can successfully meet the production requirements (e.g., meet the hard and soft constraints). The hard requirements must be satisfied, while the soft requirements can be negotiated (e.g., relaxed) within a tolerance range with penalties. A smaller, more focused clustering RA set is generated by the second match-making step:

$$\tilde{P}_q(e_{d_i}, Ag(e_{d_i}, RA_d)) \subseteq A_t(e_{d_i}, C_\ell(e_{d_i}, RA_d)) \tag{2.6}$$

where $A_t(e_{d_i}, C_\ell(e_{d_i}, RA_d))$ represents the set of resource attributes of e_{d_i} in the capability models of the clustering RAs. $\tilde{P}_q$ defines the production requirements with relaxed soft requirements. As such, the RAs that satisfy Eqn. (2.6) form a new cluster for RA_d with respect to the affected event e_{d_i}:

$$\begin{aligned} \tilde{C}_\ell(e_{d_i}, RA_d) = \{RA_c \mid RA_c \in C_\ell(e_{d_i}, RA_d), \\ \tilde{P}_q(e_{d_i}, Ag(e_{d_i}, RA_d)) \subseteq A_t(e_{d_i}, RA_c)\} \end{aligned} \tag{2.7}$$

The RAs in the cluster $\tilde{C}_\ell(e_{d_i}, RA_d)$ represent the subset of RAs that can perform e_{d_i} and satisfy the production requirements of e_{d_i}.

Schedule generation Once the cluster $\tilde{C}_\ell(e_{d_i}, RA_d)$ is formed, each RA in the cluster follows the same process to generate a new schedule. For simplicity, the following description focuses on one RA_c in the cluster $\tilde{C}_\ell(e_{d_i}, RA_d)$. As mentioned in Eqn. (2.2), a new event sequence, s_{new}, needs to be formed to achieve the transitions from x_{prior} to x_{post} in terms of location and physical composition. However, as defined in Section 2.2.1, an event can only achieve either a location or physical composition transition. Therefore, the clustering RAs must verify whether the event e_{d_i} satisfies the production needs given in Eqn. (2.2) or if other events will be needed.

The RAs are grouped into two classes in Section 2.1.2. If RA_d is a transportation RA, then e_{d_i} must be an event that drives a location change of the product and does not change the physical composition (i.e., $x^c_{prior} = x^c_{post}$). In this case, Eqn. (2.2) is rewritten as:

$$Tr(x_{prior}, e_{d_i}) = x_{post}, \text{ with } x^c_{prior} = x^c_{post} \tag{2.8}$$

For location events, a single clustering RA can generally replace RA_d without the need for further RA coordination to form a feasible new schedule. In this example, the new schedule s_{new} only contains e_{d_i}.

However, in the case where RA_d is a transformation RA, the sequential events associated with e_{d_i} that provide location transitions must be reassigned. Therefore, simply replacing RA_d with a clustering RA that performs e_{d_i} will not fulfill the required transitions in Eqn. (2.2) and other events must be included in s_{new}. Using Eqn. (2.3), the clustering RA can only drive a change in the physical composition by performing event e_{d_i}:

$$Tr(x^c_{prior}, e_{d_i}) = x^c_{post} \tag{2.9}$$

To truly replace RA_d, the transformation clustering RA will require help from transportation RAs to move the product into and out of its location, denoted by $x^\ell_{RA_c}$. Thus, transportation events that drive location changes from x^ℓ_{prior} to $x^\ell_{RA_c}$ and $x^\ell_{RA_c}$ to x^ℓ_{post} need to be found. As shown in Fig. 2.5, the clustering RA sends requests to its collaborative RAs. These collaborative RAs check their capability models and search for transportation events that will satisfy the location change requirements.

If the event does not exist, a propagation method can be used [51] to find more transportation events to drive the location change from x^ℓ_{prior} to $x^\ell_{RA_c}$ or $x^\ell_{RA_c}$ to x^ℓ_{post}. These events should be appended to s_{new} to form the final schedule.

Once the event sequence s_{new} is determined, the timing to perform these events needs to be determined. In previous work [11], new events were assigned to the corresponding RA without changing the existing schedule, which led to a large delay in the product cycle time. In this section, a function H is defined to calculate the earliest available time to start an event e based on the idle time I of the resource and the requested start time of e, denoted by t. Therefore, the function H serves as a heuristic that minimizes the completion time of the new event. To simplify the time propagation, the transportation is handled by adding a time interval δ between any two operations

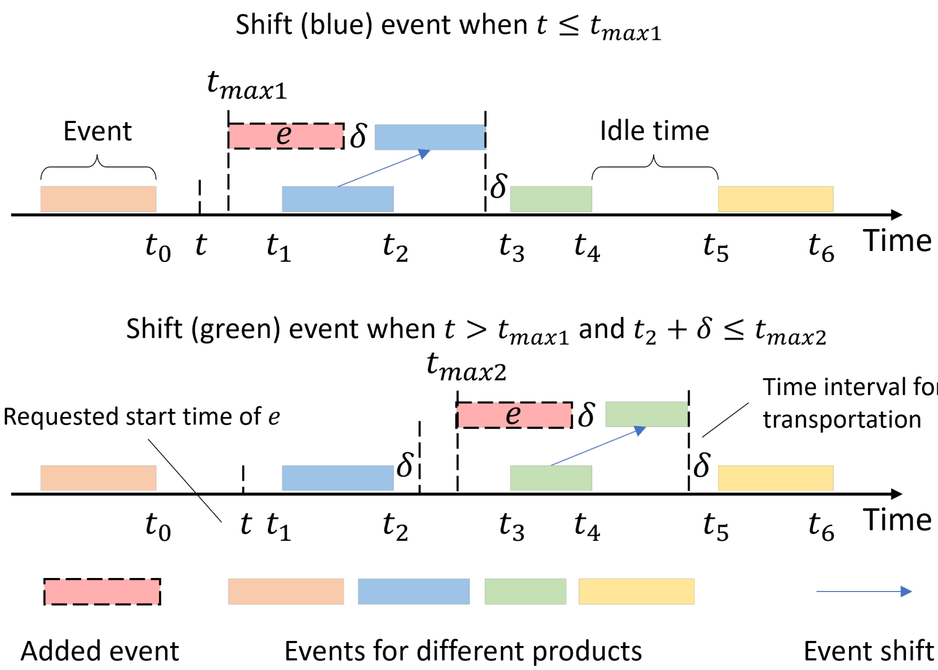

Figure 2.6: The Gantt charts that show the schedule of a clustering RA to illustrate how the function H allows one event shift

in a machine schedule. To minimize the effect on other events and products, only one scheduled operation is allowed to be shifted when adding e to a clustering RA. This constraint determines the latest time $t_{max1} = t_3 - \delta - (t_2 - t_1) - \delta - T(e)$ event e can be assigned to the current idle time interval, as shown in the upper Gantt chart in Fig. 2.6. Note that $t_{max1} = \infty$ if t_1, t_2 or t_3 does not exist. If the requested time t is larger than t_{max1}, event e cannot be scheduled before the event in (t_1, t_2). If this occurs, the clustering RA will check the next idle time interval to evaluate if $t_{max2} = t_5 - \delta - (t_4 - t_3) - \delta - T(e)$ will provide sufficient time to assign event e to this next idle time interval, see lower Gantt chart in Fig. 2.6. Note that $t_{max2} = \infty$ if t_5 does not exist. The function H is defined as follows:

$$H(I, t, \delta, e) = \begin{cases} t_0 + \delta, & t \leq t_0 + \delta \leq t_{max1}, \\ t, & t_0 + \delta < t \leq t_{max1}, \\ t_2 + \delta, & \max\{t, t_0 + \delta\} > t_{max1} \text{ and } t_2 + \delta \leq t_{max2} \end{cases} \tag{2.10}$$

where the resource idle time set, $I = \{[t_0, t_1], [t_2, t_3], ...\}$, is obtained from the $T_s(s(RA_j))$ in the resource schedule, as mentioned in Sec. 2.1.2. Note that function H can be expanded if more than one scheduled operation is allowed to be shifted.

For each new event sequence $s_{new} = e_1 e_2 ... e_s$, the post event should always start after the prior event ends:

$$t_{s,i} + T(e_i) \leq t_{s,j},\ 1 \leq i < j \leq s \tag{2.11}$$

where $t_{s,i}$ and $T(e_i)$ represent the start time and time cost (e.g., cycle time) of event e_i, respectively. The start time of a later event $t_{s,i+1}$ is obtained from function H with the requested start time of $t_{s,i} + T(e_i)$. Note that in the time interval $[t_{s,i} + T(e_i),\ t_{s,j}]$, a part remains with the current RA that performs e_i until the RA that performs the next event e_j is available. The combination of the clustering and collaborative RAs form a set of new event sequences, denoted by s_{new}, to replace s_d in order to achieve the transition from x_{prior} to x_{post}.

2.3.2 Decision-making with Risk Assessment

2.3.2.1 Risk assessment

When RAs send a response to form a new schedule, the information in the response may contain uncertainties. In this work, we define uncertainty as information about a resource attribute or state that may be stochastic or probabilistic rather than deterministic. These uncertainties can be modeled by utilizing the manufacturing data. For example, a Gaussian distribution may be used to model uncertainty in machine operation time [23]. Uncertainties introduce a potentially costly effect during the decision-making process for the rescheduling problem. We define the effects associated with variations in the rescheduling process, such as cycle time delay and schedule deviation, as risks in this section.

To consider risks in the decision-making process, all new schedules should incorporate a risk assessment process based on the set of resources chosen to replace the affected event sequence s_d. There are two key risks considered in this work:

- R_1: the risk of a new event in RA_j causing operational delays for the other products associated with this resource

- R_2: the risk of an added event in RA_j increasing the probability of breakdown

The quantification of the two risks is discussed below through an example. Note that the definition and quantification of risks, uncertainties, and how they are related may vary according to how a different resource may evaluate the risks.

Risk of an added event causing operation delays for other products scheduled with RA_j
Although event start and end times are provided in a new schedule (see Section 2.3.1.2), the actual times of these events may be shifted slightly due to uncertainties in the operation times of the events. Note that the operations before the added event are assumed to have been completed. As shown in Fig. 2.6, if the added event (red block with dash outline) takes longer to finish, the next event (blue block) for this resource could be impacted, which may also affect the following event (green block). Therefore, this risk evaluates the likelihood that a posterior event will be affected.

As defined in Section 2.3.1.2, without considering uncertainties, t_{max1} and t_{max2} represent the latest start times for which an event can be added into the sequence without affecting a posterior event for a given resource. The start time of the added event, $H(I, t, \delta, e)$, is obtained from Eqn. (2.10). If $H(I, t, \delta, e)$ is close to t_{max1} or t_{max2}, the risk of causing a delay for the following event is high. To evaluate this risk, the time deviation, denoted by Δt, between the start time and t_{max1} or t_{max2} is calculated:

$$\Delta t = \begin{cases} t_{max1} - H(I, t, \delta, e), & H(I, t, \delta, e) = t_0 + \delta \\ & \text{or } H(I, t, \delta, e) = t \\ t_{max2} - H(I, t, \delta, e), & H(I, t, \delta, e) = t_2 + \delta \end{cases} \tag{2.12}$$

If we assume a Gaussian or uniform distribution for the cycle times of different events, then t_{max1}, t_{max2}, and thus Δt are all random variables with known distributions.

Note that Δt is non-negative, where a larger Δt provides better tolerance to operation time uncertainty, which translates to a lower risk of causing delay. If there are no posterior events, then t_{max1} and $t_{max2} = \infty$, hence $\Delta t = \infty$, and the risk is zero. If $\Delta t = 0$, then the new event is scheduled to start at t_{max1} or t_{max2}. Given the uncertainty in cycle times, this indicates a high-risk decision. In a new event sequence, $s_{new} = e_0 e_1 ... e_s$, each event is added to the schedule of the specified resource. We define the risk for a given resource RA_j in the new schedule as follows:

$$Q(RA_j) = 1 - \mathbb{E}\left(\frac{\Delta t}{t_{max}}\right) \tag{2.13}$$

where $\mathbb{E}$ represents the expected value, and $t_{max} = t_{max1}$ or t_{max2} depending on the conditions in Eqn. (2.12). Note that as the difference between the maximum threshold and true start time increases, the risk goes down. Equation (2.13) limits the value of $Q(RA_j)$ to lie between 0 and 1. This type of risk can be calculated for each event within the new schedule s_{new}. The total value for Risk 1 associated with this schedule is defined as the maximum value among the resources that perform the new schedule:

$$R_1 = \max\{Q(RA_j)\}_{j \in [0,s]} \tag{2.14}$$

Risk of an added event in RA_j increasing the probability of breakdown If a resource in the new schedule breaks down, it will lead to more rescheduling requirements for the products that have uncompleted scheduled operations by this resource. Therefore, risk 2 is evaluated by determining the probability of breakdown if the resources in the new schedule are assigned new events to perform. This risk is based on the assumption that every resource has a historical Mean Time Between Failures (MTBF), and that the addition of a new event will introduce more wear and tear to the resource and move the resource closer to the MTBF.

We define the probability of resource breakdown as a function that maps RAs to a value between 0 and 1: $P_r : RA \to [0, 1]$. This probability is given as:

$$P_r(RA_j) = \frac{o_c}{o_n} \tag{2.15}$$

where o_c is the number of operations that the resource has performed since the last maintenance event, and o_n represents the nominal number of operations that the resource generally performs between breakdowns.

In a new event sequence, $s_{new} = e_0 e_1 ... e_s$, the breakdown of any resource in the new schedule makes the new schedule unsuccessful. Therefore, Risk 2 is defined as the maximum probability of the breakdown of a resource in the new schedule:

$$R_2 = \max\{P_r(RA_j)\}_{j\in[0,s]} \tag{2.16}$$

Note that as the probability of resource breakdown increases, the risk goes up. The risk value is between 0 and 1 since it is a probability calculated by Eqn. (2.15).

Since a system might apply different importance levels to the different risks, the overall risk assessment value that will be incorporated into the decision-making process is a weighted sum of the risks (R1 and R2 in this work) $\sum_{i=1}^{k} w_i R_i(s_{new})$, where w_i is the weight factor for R_i.

2.3.2.2 Decision-making

Determine the new schedule Once the risk assessment is completed, RA_d is responsible for choosing a new schedule from the set of possible event sequences S_{new}. Note that every event sequence in S_{new} satisfies all the constraints to achieve the production goal due to the proposed problem formulation and agent coordination. Therefore, the new schedule selection problem is reduced to obtain the new schedule that optimizes the rescheduling objectives defined by the manufacturer. This type of optimization can be easily solved by some classical algorithms, such as bubble sort and divide-and-conquer. Note that this optimization provides the optimal new schedule from the candidate solution set S_{new}. However, the global optimal solution may not be in the candidate set S_{new} since all the candidate new schedules are formed by agent local decision-making. An example is given in Eqn. (2.17):

$$s^*_{new} = \underset{s_{new} \in S_{new}}{\arg\min} \; \mathcal{J}(s_{new}) \tag{2.17}$$

where $s^*_{new} \in S_{new}$ is the event sequence that provides minimal objective. The multi-objective function $\mathcal{J}$ is a sum of the cost, penalty, and risk for one event sequence $s_{new} = e_1 e_2 ... e_s$, as shown in Eqn. (2.18):

$$\mathcal{J}(s_{new}) = \sum_{i=1}^{s} \boldsymbol{\alpha} \boldsymbol{C}(e_i) + \sum_{i=1}^{s} \beta_i p_i^a + W \sum_{i=1}^{k} w_i R_i(s_{new}) \tag{2.18}$$

where $\boldsymbol{C}(e_i) = [C_1(e_i)\ C_2(e_i)\ \cdots\ C_n(e_i)]^T$ captures a nominal cost function for event e_i based on n metrics and $\boldsymbol{\alpha} = [\alpha_1 \alpha_2 \cdots \alpha_n]$ describes the corresponding weights. The pre-defined cost metrics could include operation time, finish time, energy cost, resolution, etc. If there are soft constraints that must be negotiated, p_i^a denotes the penalty for performing e_i and β_i is the corresponding weight. The risks associated with the given sequence are evaluated in $\sum_{i=1}^{k} w_i R_i(s_{new})$ for a given sequence. Parameter W is used to scale the risk based on the scale of the cost and penalty and what value the decision maker places on the assessment of risk. Note that in Eqn. (2.18), the objectives, penalties, and risks are defined by the manufacturers and the weight parameters depend on how the manufacturers desire to balance the objectives, penalties, and risks. Future work will investigate the sensitivity of Eqn. (2.18) to changes in these values and identify a method for optimizing under various conditions.

As shown in Fig. 2.5, the affected resource, RA_d, informs the RAs that will be associated with the new event sequence, s^*_{new}, through a Communication Manager. The new RAs receive the notification and pass the information to their Knowledge Bases to update their resource schedules and provide high-level control for their associated physical resources to perform the events.

No schedule found The result of the rescheduling problem depends on resource redundancy and available capacity, which are the assumptions we made in Section 2.1.2. In practice, manufacturing resources are limited in a factory, therefore, the existence of a feasible new schedule is not guaranteed. Therefore, in the proposed method, if no schedule is found within the required constraints (e.g., no redundant resources available), the RA_d will request the central controller of the manufacturing system or human manager to make further decisions or relax additional constraints.

As mentioned in Sec. 2.1.2, the centralized method evaluates all of the resources in the system for each event that needs to be replaced during the rescheduling process. Therefore, the centralized method forms the candidate solution set S_{new} by considering all the combinations of resources in the system and then solves the following optimization:

$$\min_{s_{new} \in S_{new}} \quad \mathcal{J}(s_{new}) \tag{2.19a}$$

$$\text{s.t.} \qquad Tr(x_{prior}, s_{new}) = x_{post} \tag{2.19b}$$

$$t_{s,i} + T(e_i) \leq t_{s,j},\ 1 \leq i < j \leq s, e_i \in s_{new}, \tag{2.19c}$$

where these objectives and constraints are the same as those considered by the distributed method.

2.4 Case Studies

To evaluate the feasibility and performance of the proposed framework, the proposed RA architecture and rescheduling strategy are implemented in a simulated manufacturing system. In this section, the setup of the simulated manufacturing system and the results of the case study are provided.

2.4.1 Case Study Set-up

In this study, we use a Repast Symphony (RepastS) platform [99] to model a multi-agent system and simulate the behavior of the agents due to its flexibility and scalability properties. The simulated manufacturing system represents a modified version of the Intel Mini-Fab [100], a semiconductor manufacturing facility. The simulated system contains two infinite-sized buffers, Entry

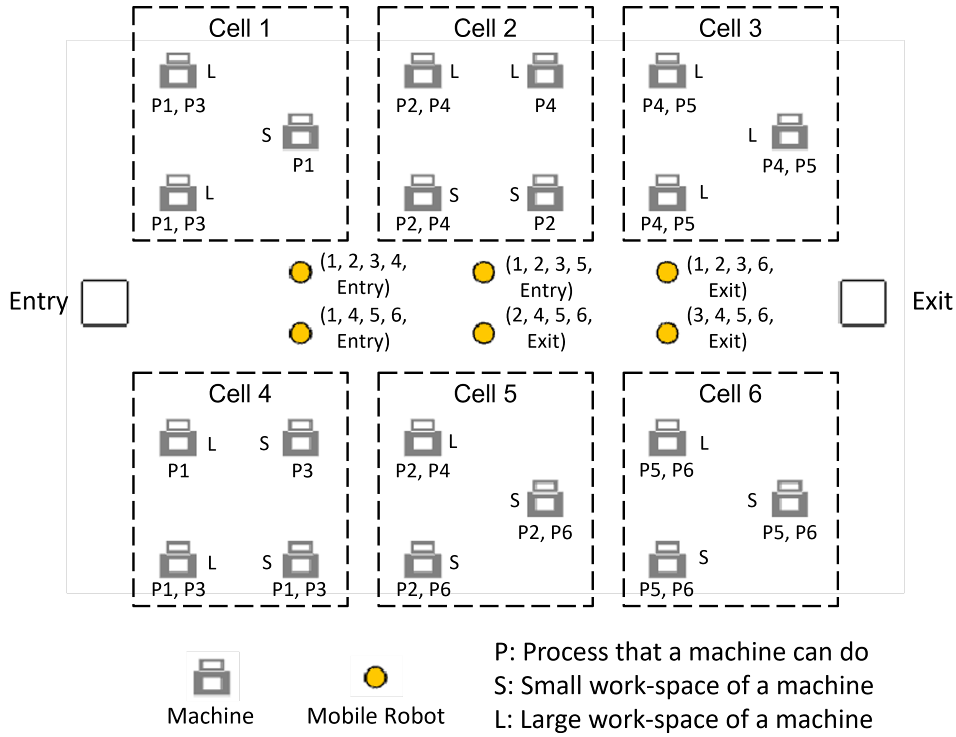

Figure 2.7: A screenshot of the facility layout from the RepastS environment. The annotations indicate the capability of each resource.

and Exit, and 20 machines that are connected via a network of 6 mobile robots, as shown in Fig. 2.7. The annotations represent the capabilities of the machines and mobile robots. There are 6 different processes (P1-P6) that the machines can perform, where the operation costs in ticks (RepastS unit of time) range from 110-200. For example, the annotation for a machine indicates which processes it can perform and whether the workstation space is large or small. The annotation for a mobile robot represents which cells and buffers it can reach when moving the products.

Two types of products, labeled S (small) and L (large), are introduced into the system, where each type of product has the following process requirements:

- S-product: P1 $\rightarrow$ P2 $\rightarrow$ P3 $\rightarrow$ P6
- L-product: P1 $\rightarrow$ P3 $\rightarrow$ P4 $\rightarrow$ P5

Machines labeled L can operate both L-products and S-products, while machines labeled S can only operate on S-products. Products enter the system from the Entry buffer and leave the system through the Exit buffer after completing the desired processes.

2.4.2 Case Studies: Rescheduling for Machine Breakdowns

In this simulated manufacturing system, 50 L-products and 50 S-products are fed alternatively into the system with a pre-generated initial production schedule. Products enter the facility every 30 ticks starting at tick 10. To provide an opportunity for a rescheduling event to occur, the initial production schedule is designed with 50% resource utilization. Uncertainty in machine operation time and the probability of machine breakdown are added to all machines in the simulated system. The system starts operations with the probability of machine breakdown ranging from 3.3% to 10%. If a machine undergoes a breakdown, a rescheduling process will be triggered. The mean time to repair ranges from 1000-1500 ticks for a broken machine. Note that if the breakdown occurs when the machine is processing a product, the product will be damaged and cannot be recovered. The rescheduling decision-making considers the completion time of s_{new} as the objective C, and this case study does not have soft constraints, thus $\alpha = 1$ and $p_i^a = 0$. We conduct two case studies to evaluate the performance of the proposed distributed method.

2.4.2.1 Centralized versus distributed

The first case study aims to evaluate the performance trade-offs between the centralized method and the proposed distributed method in terms of optimal cycle time and computational efforts. We run two simulation scenarios where the system uses centralized and distributed methods respectively as the rescheduling decision-making strategy. Note that risks are not included in this case study since it does not affect these trade-offs. For each scenario, we run 5 trials to evaluate their performance with the following metrics:

- Cycle time
- Number of agent communications
- Running time of the decision-making implementation

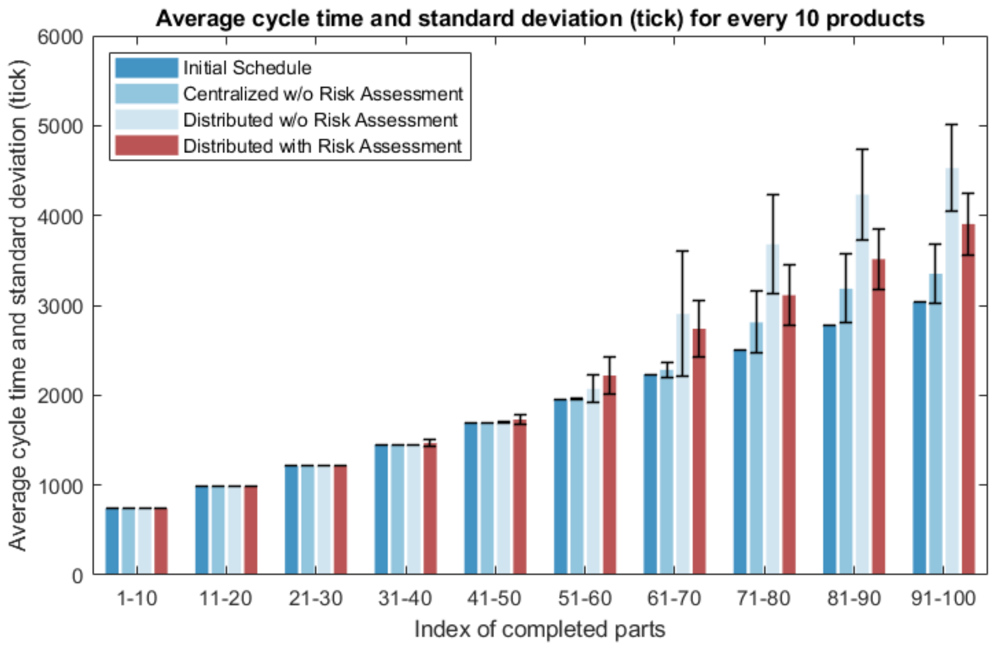

Figure 2.8: Average cycle time and standard deviation for every 10 products for 5 trials in different scenarios.

The number of communications in the centralized method includes the request for rescheduling, the requests to and responses from all the RAs in the system to collect information, and the notifications to the agents whose production schedules need to change. Therefore, each rescheduling process requires $r \times \sum_{e_{d_i} \in E_d} |s_{d_i}|$ communications, where r is the number of RAs in the system. In the distributed method, the number of communication includes all agent requests, responses, and inform messages, as defined in Sec. 2.3.1.2. The communication only occurs within local clustering RAs and their collaborative RAs (i.e., a subset of all the RAs in the system), thus the distributed method requires less communication, as showcased in Table 2.2.

Figure 2.8 shows the product cycle time in different scenarios. The centralized method reoptimizes the whole system to generate a new schedule with a shorter cycle time than the distributed method. However, as shown in the first part of Table 2.2, the centralized method requires more communication and larger computational efforts to reschedule the system. In practice, more communication potentially leads to a larger information delay, thus the centralized method lacks the ability to respond to disruptions dynamically and quickly. Furthermore, the computational ef-

Table 2.2: Performance evaluation of the manufacturing rescheduling process in different trials in terms of certain metrics

Metrics	Scenarios	Values in 5 trials					Ave.	Pct.
Centralized versus distributed								
# of agent communication	Centralized	1430	1155	1650	1100	1320	1331	N/A
	Distributed	1053	858	1248	780	1014	991	N/A
Total running time (sec) of rescheduling processes	Centralized	10.1	18.7	16.4	8.8	9.4	12.7	N/A
	Distributed	0.22	0.18	0.34	0.49	0.33	0.31	N/A
With risk assessment versus without risk assessment								
# damaged products	W/ risk assessment	3	5	3	3	5	3.8	3.8%
	W/o risk assessment	9	7	8	6	5	7.0	7.0%
# broken machines	W/ risk assessment	6	7	4	5	7	5.8	29%
	W/o risk assessment	11	10	10	8	9	9.6	48%
# rescheduled processes	W/ risk assessment	17	20	7	14	23	16.2	4.05%
	W/o risk assessment	32	25	35	21	31	28.8	7.20%
Peak risk values	W/ risk assessment	0.26	0.26	0.27	0.26	0.27	0.27	N/A
	W/o risk assessment	0.27	0.29	0.28	0.29	0.29	0.28	N/A
Average risk values	W/ risk assessment	0.23	0.22	0.21	0.22	0.22	0.22	N/A
	W/o risk assessment	0.25	0.26	0.24	0.24	0.25	0.25	N/A

forts of the centralized method increase as the size and complexity of the set-up increase in scale. In this case, the distributed method can provide advantages by using local communication to reduce communication and computation time.

2.4.2.2 With risk assessment versus without risk assessment

To further investigate how the introduction of risk assessment affects the rescheduling decision-making and the system performance, we run 5 trials where the system uses the distributed method with the incorporation of risk assessment into the rescheduling decision-making process. As discussed in Sec. 2.3, the overall risk associated with a new schedule is calculated as the weighted sum of two risk factors, $w_1 * R_1 + w_2 * R_2$. In this case study, we simulated a scenario where the manufacturer cares more about machine breakdowns, thus we selected $w_1 = 0.2$ and $w_2 = 0.8$. Note that different risks and weights can be defined and chosen, while the performance under different parameters can be investigated in future work. The weighting gain, W, for the total risk value has been selected to assign value to the introduction of risk and scaled to ensure comparable unit values. Note that for this example, the mobile robots are assumed to be reliable and are not considered within the risk assessment at this time.

Besides cycle time, we introduce the following metrics to evaluate the system performance with risk assessment:

- Number of damaged products
- Number of broken machines
- Number of rescheduled processes
- Peak and average risk values of the new schedule

The results are shown in the second part of Table 2.2, which indicates that system performance with and without risk assessment varies across the different trials. With risk assessment, the average number of damaged products is 3.8 versus an average number of 7 without risk assessment. When combined with the results from the number of machine breakdowns (5.8 versus 9.6), these results illustrate how the consideration of risk results in a rescheduling strategy that selects a less risky schedule that reduces the potential for machine breakdowns and damaged products. Note that breakdowns may occur while the machines are not processing, thus the number of broken machines is larger than the number of damaged products.

To investigate how the rescheduling strategy impacts the potential for machine breakdown and the trigger of a new rescheduling task, the number of rescheduled processes is also presented in Table. 2.2. On average, when risk assessment is included, the rescheduling process is triggered 16.2 times, while it is triggered 28.8 times when risk assessment is ignored. Peak and average risk values provide a measure of the associated risks inherent in the two strategies. Note that when risk is included in the cost function, the decision-making strategy results in a selection process that chooses the event sequences with lower risks (0.27 versus 0.28 peak and 0.22 versus 0.25 average risk values).

To show how the assessment of risk affects the completion of products within the simulated facility, the mean values and the standard deviations of the average cycle time for every 10 products for the 5 trials are shown in Fig. 2.8. As Fig. 2.8 shows, the first 40 products have nearly identical cycle times. Although these products might be in the system during a later breakdown event, the risk of machine breakdown and a rescheduling event is low during this initial period.

Interestingly, the impact of risk assessment really becomes apparent during the 61-70 part completion set. At this point in the simulation, the risk for machine breakdown is increasing as machine usage time gets closer to the MTBF for a given resource. Once R2 begins to increase, the decision to select the less risky event sequence results in fewer machine breakdowns, less rescheduling, and a lower average cycle time. This trend continues, with considerable variability beginning to be introduced into the cycle times as illustrated in Fig. 2.8.

Overall, the consideration of risk into the event sequence decision results in fewer damaged products and broken machines, a reduction in the number of triggered rescheduling processes, and an improvement in the system throughput as compared to the decision strategy that does not consider risk. These results showcased that incorporating risk assessment affects the agent decision-making in the rescheduling process. Our results indicate that the introduction of the risk assessment value resulted in a smaller number of broken machines and damaged products, as well as a reduction in cycle time variability.

2.4.3 Discussion and Insights

The case study has showcased the feasibility and performance of the proposed multi-agent framework, specifically demonstrating how risk assessment affects agent decision-making. However, there are other aspects that may affect the framework performance, which will be investigated in future work. Firstly, the framework can be easily adapted to different case study setups. As the set-up scales down, the set of candidate solutions might shrink. Thus, there might be no big difference between centralized and distributed methods in terms of needed communications. Besides, the risk assessment may not affect decision-making significantly since the choices are limited. On the other hand, as the size and complexity of the set-up increase in scale, both the size and the variety of the candidate solutions might grow. Therefore, the risk assessment can make a big difference in the selection of the new schedule.

Based on the objectives, risks, and parameters used in the simulation, this case study simulated a scenario where the manufacturer cares about cycle time and machine breakdown. The results indicate that the proposed method reduced the production cycle time and machine breakdowns, which showcased the feasibility and performance of the proposed method. Therefore, different objectives, risks, parameters, and metrics can be used in the proposed method, while it inevitably might change the results toward a better or worse direction. An enhanced understanding of the sensitivity of the parameters design, such as identifying the set of conditions under which the algorithm always outperforms other algorithms, is left for future work.

In addition, other metrics, such as makespan and machine utilization rate can also be analyzed based on the provided results. The cycle time in Fig. 2.8 can reflect the makespan since the entry time of a specific product in each trial is identical. Therefore, without risk assessment, the makespan of the schedule is larger. Note that the large makespan occurs in the case where the system produced fewer products. As a result, the machine utilization rate when the rescheduling does not consider risks is lower than the scenario when the risk assessment is incorporated.

2.5 Concluding Remarks

In this chapter, a model-based RA architecture that enables effective agent coordination and dynamic decision-making is designed. The proposed RA architecture contains a Knowledge Base, Decision Manager, and Communication Manager. Based on this architecture, a rescheduling strategy is developed to incorporate risk assessment via RA coordination in the presence of resource breakdown. The proposed work can be used to create manufacturing resource models that enable dynamic and resilient rescheduling for manufacturing systems. Implementation of the proposed framework in a simulation-based case study has been done to evaluate the effectiveness of the proposed architecture. In particular, the case study demonstrates that the proposed agent-based distributed method reduces the communications and computational efforts that are needed for rescheduling while losing some optimality in throughput compared to the centralized method. Additionally, the case study illustrates the improvement in throughput when risk is considered within the rescheduling problem. Showcased through a simulation study, the proposed work provides a more resilient and robust rescheduling distributed decision-making strategy to recover and maintain throughput performance under uncertain manufacturing environments. From the managerial perspective, manufacturers can use this work to model and monitor their factory floor as the agents store the physical information and keep it updated. Furthermore, this framework can be used as a decision support system since the agent decision-making ability can provide manufacturers with several solutions to respond to disruption depending on different objectives and parameters defined by the manufacturers.

However, it is noticed that focusing on rescheduling within the manufacturing systems may not provide satisfying solutions for external disruptions, such as customer demand and material shortage. In addition, the distributed approach offers more advantages of flexibility and agility for local disruptions in larger and more complex systems since it does not require communication and re-optimization throughout the entire system. Therefore, the following chapter will apply the proposed multi-agent framework to supply chain networks is studied.

CHAPTER 3

Multi-Agent Framework for Re-planning in Supply Chain Networks

This chapter presents a model-based multi-agent framework for agile and resilient disruption responses in supply chain networks. The related work has been published in [17] and submitted to [101]. As described in Chapter 2, rescheduling in manufacturing systems has limited flexibility for various disruptions in industrial environments. Therefore, enterprises should have the ability to re-plan both production and transportation and determine decision-making strategies based on the disruption and network attributes. However, most existing agent-based disruption reaction strategies use rule-based reasoning and pre-define a disruption database, thus the flexibility of the system is limited to the set of pre-defined scenarios. In this chapter, we focus on the re-planning problem in the presence of an agent loss in supply chain networks. Following the agent architecture described in Chapter 2, we introduce a model-based multi-agent supply chain framework that enables agent exploration and iterative communication for a dynamic and agile disruption response.

The rest of the chapter is organized as follows. A literature review and supply chain description are presented in Section 3.1. Section 3.2 describes the proposed model-based multi-agent framework for supply chains. The agent communication and decision-making for re-planning are discussed in Section 3.3. Section 3.4 presents a comprehensive case study that investigates the performance and network attributes. Concluding remarks are presented in Section 3.5.

3.1 Background and Overview

In this section, a literature review of the multi-agent approach for the re-planning problem in supply chain networks is provided. Then the proposed topological and capability descriptions for supply chain networks are presented.

3.1.1 Literature Review

A significant effort has been made to develop disruption mitigation approaches to re-plan supply chain productions and flows, leveraging both centralized and distributed decision-making strategies. Most existing literature in this domain focuses on proactive methods for supply chain disruption mitigation, such as demand forecasting, inventory management, and stochastic optimization methods that estimate potential disruptions in advance to enhance supply chain robustness [39, 40]. However, unexpected disruptive events, such as supplier loss, require enterprises to make quick and effective decisions in response to the disruption. Therefore, it is important and necessary to understand how the centralized and distributed approaches impact the recovery performance depending on the attributes of the disrupted agents [5].

To address the problem, a comprehensive supply chain description is needed. In the literature, researchers have described a supply chain as a network with vertices (e.g., supplier, customer, etc.) and edges (e.g., transportation), along with their associated attributes and parameters (e.g., cost and capacity) [17, 102, 103]. In this dissertation, both the vertices and edges we consider have intelligence and thus are defined as agents. Therefore, understanding the agents and their attributes within a supply chain network can help determine the impact of disruptions based on where disruptions occur and the critical performance metrics. From the topological perspective, existing literature has made a significant effort in conceptualizing supply chain disruptions and investigating the effects of the overall network topology on supply chain resilience and robustness [103–106]. However, the existing studies did not discuss how the agent attributes within this network (e.g., capabilities, connectivity, etc) impact the mitigation performance. Furthermore, agent capability attributes are important for understanding the impact of the disrupted agent on supply chain disruption recovery. This specification has implications for how we determine the decision-making approach for disruption mitigation.

In terms of decision-making approaches for supply chain management, centralized models are widely used to provide optimal solutions based on specific objectives (e.g., product flow cost) [20]. As discussed in Chapter 1, centralized approaches require information about the entire supply chain in order to re-optimize the system in response to a disruption [28]. As the complexity and scale of supply chains increase, it becomes more difficult to remain agile in the presence of multiple disruptions [7] due to communication demands and computational complexities that arise in these scenarios [29]. Therefore, to improve the flexibility and agility of supply chain networks, researchers have proposed multi-agent systems to conduct distributed decision-making for agile supply chain disruption mitigation [7, 32, 46, 47, 107].

In the existing literature, most agent-based disruption reaction strategies are based on pre-defined disruption scenarios and reactive actions via a stochastic programming model [67], a Petri Nets model [68], or a case-based disruption reaction database [37]. In many examples, the disruption reaction performance of these methods is limited to a set of pre-defined scenarios, and it can be difficult and even impossible to cope with unexpected disruptions outside of the pre-defined set. To address this limitation, agents need to be equipped with model-based knowledge to make decisions dynamically. However, existing multi-agent approaches focus on either system-level architectures or rule-based agents [32]. The works [9, 54, 55] provide general descriptions of agent attributes and functions at a conceptual level. References [57, 69, 70] use rule-based reasoning to guide agent decisions, while [29, 58, 69] introduce case-based agents that provide pre-planned decision-making and coordinated behaviors for the agents. In these methods, relying on a rule-based strategy limits the ability of the agents to readily adapt to unexpected supply chain disruptions. Additionally, it is difficult to scale this rule-based approach to larger and more complex systems, with the addition of more rules reducing the flexibility of agent behaviors. Therefore, a new distributed multi-agent framework must be developed to improve the flexibility and agility of supply chains in response to disruptions.

In addition, to make a proper disruption response, it is beneficial to understand how different approaches perform for different disruptions and interested objectives. In general, distributed approaches are efficient, agile, and flexible enough to react to disruptions quickly, while the solutions are often locally optimal and may not always align with the global objectives or constraints [44, 80]. In supply chain literature, comparisons between distributed and centralized decision-making focus on the initial optimization of a supply chain [108, 109] rather than the re-optimization or dynamic response of the system after a disruption occurs. Therefore, to the best of our knowledge, no study has carried out an evaluation of the performance of a centralized or distributed approach based on the attributes of the disrupted supply chain agents in a complex supply chain network.

To address this limitation, the contributions of this chapter include: (1) formulations of agent's attributes (e.g., capability, connectivity) within the context of a supply chain network; (2) development of a model-based multi-agent supply chain framework that enables agent exploration and iterative communication in response to unexpected disruptions; and (3) investigation of how network attributes of a disrupted agent affect supply-chain performance for centralized and distributed approaches through a comprehensive case study.

3.1.2 Supply Chain Overview and Formulation

3.1.2.1 Overview and assumptions

Consider a supply chain network $\mathcal{G}(\mathcal{V}, \mathcal{E})$ with $\mathcal{V}$ being the set of vertices, representing supply chain entities, such as suppliers, customers, etc., and $\mathcal{E}$ being the set of edges, representing product/material flows between the entities. The associated information (e.g., demand, production, cost, and capacity) with the vertices and edges is also included [17, 103]. In this dissertation, both vertices and edges in the supply chain network have intelligence and thus are defined as agents. The corresponding supply chain agent network $\mathcal{G}^a(\mathcal{A}, \mathcal{L})$ includes all the vertices in $\mathcal{V}$ and edges in $\mathcal{E}$ in the agent set $\mathcal{A}$, and the agents in $\mathcal{A}$ are connected by a set $\mathcal{L}$ of communication links. The following agent types are considered in the network: customer, distributor, original equipment manufacturer (OEM), tier supplier, and transporter.

This work investigates how the network attributes of a given agent (e.g., connectivity, capability) impact the performance of the supply chain system in response to a disruption. Supply chain disruptions are classified into different categories, including internal and external disruptions based on their causes, as well as supplier and customer disruptions based on their locations [39, 40]. In this study, our primary focus is on an unexpected supplier loss, which may be triggered by natural disasters or workforce strikes.

The response is defined as a new flow plan to minimize total cost and demand dissatisfaction. We explore both distributed and centralized decision-making approaches to provide a comparison. The centralized approach solves the problem from the entire supply chain network perspective:

$$
\begin{aligned}
\min_{y,x,I,p,\beta,\zeta,\Delta} \quad & \mathcal{J} = \sum_{(i,j)\in E, k\in K} c_{ijk} y_{ijk} + \sum_{i\in V, k\in K} e_{ik} p_{ik} \\
& \quad + \sum_{i\in V, k\in K} \rho^d_{ik} \Delta^d_{ik} && \text{(3.1a)} \\
\text{s.t.} \quad & \sum_{j:(i,j)\in E} y_{ijk} - \sum_{j:(j,i)\in E} y_{jik} + \sum_{k'\in K} r_{kk'} p_{ik'} \\
& - p_{ik} = x_{ik} + I^0_{ik} - I_{ik}, \ \forall i \in V, \ k \in K && \text{(3.1b)} \\
& \sum_{k\in K} y_{ijk} \leq q_{ij} \beta_{ij}, \ \forall (i,j) \in E && \text{(3.1c)} \\
& \sum_{k\in K} p_{ik} \leq \bar{p}_i \zeta_i, \ \forall i \in V && \text{(3.1d)} \\
& \Delta^d_{ik} \geq x_{ik} - d_{ik}, \ \forall i \in V, \ k \in K && \text{(3.1e)} \\
& y_{ijk}, x_{ik}, I_{ik}, \Delta^d_{ik} \geq 0, \ \zeta_i, \beta_{ij} \in \{0,1\}, \\
& \qquad \forall i \in V, \ (i,j) \in E, \ k \in K, && \text{(3.1f)}
\end{aligned}
$$

where (3.1a) are the total costs of flow, inventory, and production, as well as penalty costs of unsatisfied demand. Constraint in (3.1b) defines the flow balance of each product for each agent; constraint in (3.1c) and (3.1d) limits the flow on each edge and production at each agent by its given capacity; constraint in (3.1e) computes the unsatisfied demands of each product at each vertex; and constraint in (3.1f) specifies the domains of variables. Once the disruption is identified, a centralized decision-maker will re-run the centralized model with updated network structures, parameters, and constraints to determine the re-optimized decisions. More details can be found in our previous work [17].

To describe the proposed distributed approach, we first make the following assumptions to specify our scope:

A.1 Supply chain agents have self-awareness of their own attributes and can communicate and make decisions.

A.2 The supply chain network or system contains supplier redundancy and operates within the capacity limit of the initial plan.

A.3 Unexpected disruptions are in the form of a lost agent (e.g., the agent is unable to perform their set tasks) within the supply chain network. This disruption can be detected by the associated agent.

A.4 An agent retains communication capabilities even in the presence of a disruption.

A.1 defines the agents' abilities to make individual or locally dependent decisions in response to a disruption. A.2 ensures that a new product flow plan can be determined. A.3 guarantees that the disruption will be identified by the agent if it occurs and also designates how the supply chain network will be impacted by the disruption. A.4 is necessary to enable local negotiations in response to the disruption.

To understand how disruptions affect supply chains, we provide detailed supply chain descriptions at the network and agent levels. Specifically, we focus on the role of each agent in the supply chain network from both topological and capability perspectives. Table 3.1 summarizes the notations used in this section. We also illustrate key definitions in Figure 3.1.

Table 3.1: Nomenclature for multi-agent supply chain networks

Supply chain description	
$\mathcal{G} = (\mathcal{V}, \mathcal{E})$	supply chain network (vertices and transportation edges).
$\mathcal{G}^a = (\mathcal{A}, \mathcal{L})$	agent network (agents and communication links).
K	set of product types.
y_{ijk}	units of product k transported from agent a_i to a_j.
f	network flow state including all product flows.
$m = (o, k)$	capability of performing operation o for product k.
$\mathcal{Z}(k)$	the set of needed product types to make a product k.
Agent attributes	
$\mathcal{C}_i$	connectivity of agent a_i.
$\mathcal{D}_i$	depth of agent a_i in the network.
$\mathcal{R}_i(m)$	capability redundancy of agent a_i for capability m.
$\mathcal{P}_i$	production complexity of agent a_i.
Agent communication and decision-making	
a_e	disrupted (i.e., lost) agent
y_e^0	all the initial product flows related to a_e.
$\mathcal{A}_{dm}$	set of demand agents
d_{ik}	units of product k that agent a_i needs.
Δf	changes of network flow state.
$\mathcal{M}_j(k)$	set of agents that a_j sends requests to for product k.
$\bar{y}_{ijk}$	maximum units of product k that agent a_i determines to provide to a_j.
$\hat{y}_{ijk}$	units of product k that agent a_j determines to get from a_i.
Metrics for performance evaluation	
O	sum of the costs for transportation and production that exceed the nominal agent capacity.
N_c	sum of modified edges (e.g. type and/or amount of production flow) and agents (e.g. new production volume or capability).
N_a	sum of additional edges and agents needed for transportation and production.
M	the number of agent communication exchanges used to derive a response to disruption.

3.1.2.2 Supply chain attributes

Based on a topological description from literature [102] as well as agent attributes introduced in [17], we describe the role of an agent in a supply chain network from topological and capability perspectives. In the following description, we will use Wheel and Rim agents from Figure 3.1 as examples.

Connectivity Defined as the number of in-flow and out-flow edges (i.e., transportation units) to or from agent a_i:

$$\mathcal{C}_i = \sum_{a_j \in V} b_{ij} + b_{ji}, \tag{3.2}$$

where $b_{ij} = 1$ if edge (i, j) is associated with material or product flow from a_i into a_j, and 0 otherwise. From Figure 3.1, the connectivity of the Wheel agent is given as $\mathcal{C} = 3 + 2 = 5$. The *Connectivity* of an agent represents the number of other agents that will be affected if this agent is disrupted.

Depth Defined as the maximum number of edges between a_i and the final layer (e.g., customer)

$$\mathcal{D}_i = \max_{a_j \in \text{Customer}} d(a_i, a_j), \tag{3.3}$$

where $d(a_i, a_j)$ is the geodesic distance, defined as the minimal length of a path between two agents a_i to a_j. In Figure 3.1, assuming the store represents the customer layer, the depth of the Wheel agent is found to be 1. The *Depth* of an agent represents where the agent is located in the supply chain, thus it reflects the possible ripple effect if the agent is disrupted.

Redundancy From a capability perspective, the redundancy of a_i is defined as the number of alternative agents (excluding a_i) that can perform the same capability as a_i in the agent network:

$$\mathcal{R}_i(m) = |\{a_j | a_j \text{ with capability } m, a_j \in \mathcal{A} \setminus \{a_i\}\}|, \tag{3.4}$$

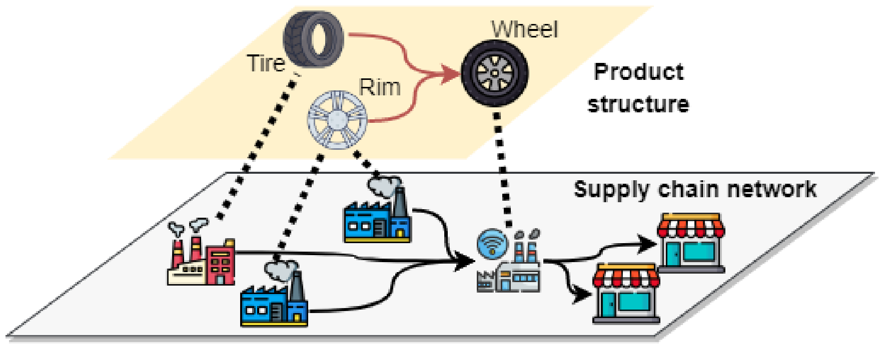

Figure 3.1: An example of a supply chain network and product structure used to illustrate the proposed supply chain descriptions.

where m represents a specific capability of agent a_i. A detailed definition of m is given in Section 3.2.1. This attribute indicates the number of backup suppliers for a given capability. In Figure 3.1, the network contains two agents that can produce the Rim; thus, for each Rim agent, the capability redundancy is $\mathcal{R}_i(m) = 1$. The *Redundancy* of an agent indicates whether there are backup agents to recover the product flows if this agent is disrupted.

Complexity Defined as the sum of final product types that require products from agent a_i and the material/component types necessary for a_i's production:

$$\mathcal{P}_i = |\bigcup_{k \in K_i} \{k_f \mid k \in \mathcal{Z}(k_f), \forall k_f \in K_f\}| + |\bigcup_{k \in K_i} \mathcal{Z}(k)|, \tag{3.5}$$

where K_i represents the set of product types that a_i can produce and K_f represents the set of final product types in the supply chain. The function $\mathcal{Z} : K \to 2^K$ maps a specific product type k to the set of components and/or materials that are needed to produce k. We denote K as the set of all product types in the supply chain. In Figure 3.1, the production of the wheel requires a tire and a rim, $\mathcal{Z}(\text{Wheel}) = \{\text{Tire, Rim}\}$. It is assumed that there is only one product type that requires the wheel. Thus, the production complexity of the wheel agent $\mathcal{P}_{wheel} = 1 + 2 = 3$. The *Complexity* of an agent represents the number of product types that will be affected if this agent is disrupted.

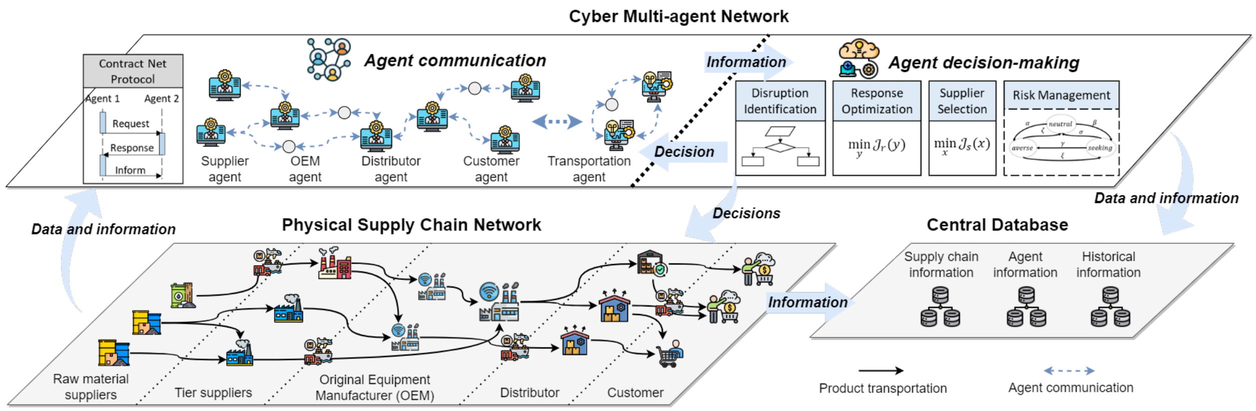

Figure 3.2: The proposed multi-agent framework for supply chain disruption mitigation.

3.2 Multi-Agent Framework

In this section, we investigate a multi-agent framework to deploy distributed decision-making for supply chain disruption mitigation. Figure 3.2 provides a high-level overview of the proposed multi-agent supply chain framework, including a physical supply chain network, a cyber multi-agent network, and a central database. The physical supply chain network contains the business entities and product flows from the supply chain in the real world. The multi-agent cyber network consists of an agent communication layer and an agent decision-making layer. Each agent is a cyber representation of its physical counterpart and will be initialized with its own version of agent architecture, as shown in Figure 3.3. The agents obtain information from the physical supply chain and communicate with each other to share the information. Based on their own knowledge and shared information, agents are able to make their own decisions, such as supplier selection, and command the decided changes to the corresponding physical entity, as shown in Figure 3.2. The central database stores all the information from the physical supply chain and cyber agent networks. In this section, we provide our design of a supply chain agent architecture and describe how agents communicate and make decisions for disruption mitigation.

The design of supply-chain agents is based on the architecture of our previous architecture for manufacturing agents [66], which consists of three modules: a Knowledge Base, a Decision Manager, and a Communication Manager. The Communication Manager is identical since it serves

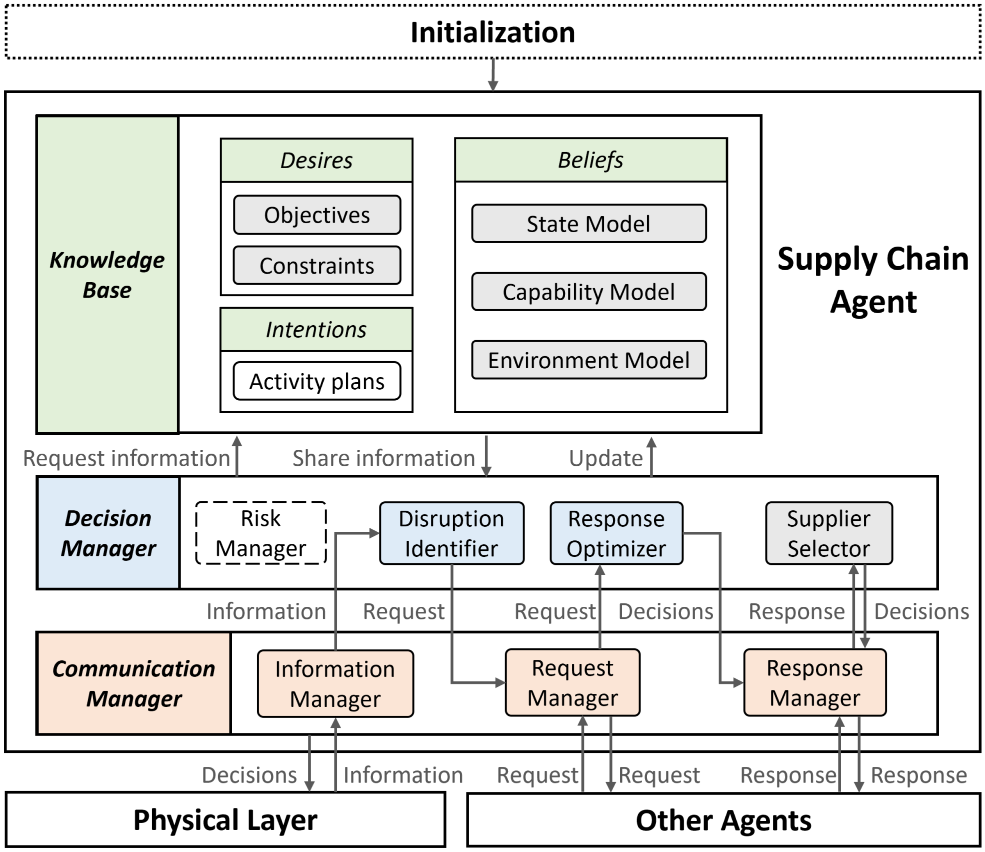

Figure 3.3: The proposed supply chain agent architecture.

as an interface for information exchange. However, the Knowledge Base and Decision Manager are more agent-specific and require new formulations. Figure 3.3 depicts a detailed design of the proposed agent architecture, including specific components in the modules and component-to-component information exchange.

3.2.1 Knowledge base

Same as the RA in Chapter 2, the Knowledge Base of supply-chain agents is also designed following BDI architecture. Compared to the Knowledge Base of manufacturing agents, we keep the same type of knowledge (e.g., capabilities) included in agent beliefs, desires, and intentions for supply-chain agents. However, the models describing the knowledge are different and have to be re-formulated.

3.2.1.1 Beliefs

Agent beliefs, including state, capability, and environment models, represent what the agent knows about itself and its environment. These models are dynamically updated (i.e., extended, shrunk, and revised) as the agent and its environment change.

Capability model This model describes the operational behaviors that an agent can perform in the supply chain network. The capability model consists of capability knowledge and several associated mapping functions. The capability knowledge is a set of capabilities $M_c = \{m_0, m_1, ...\}$. Each capability $m_j = (o, k)$ is a tuple, where o is one of the operational behaviors, including production, inventory, or transportation, and k is a product type. Along with this high-level capability knowledge, several mapping functions are used to describe the characteristics of a capability, such as cost and capacity. The capability model is built when the agent is initialized based on its associated physical entity. This information is dynamically updated as the supply chain environment changes, such as cost increases, production line changes, etc.

Environment model An agent's knowledge of other agents in the network is encapsulated in the environment model. Based on the agent network $\mathcal{G}^a = (\mathcal{A}, \mathcal{L})$, we can describe the local communication network for a single agent $a_i \in \mathcal{A}$ as $\mathcal{G}_i^a = (\mathcal{A}_i, \mathcal{L}_i)$, where $\mathcal{A}_i \subseteq \mathcal{A}$ is a subset of agents that a_i can communicate with via links in $\mathcal{L}_i$. These agents are grouped into several subsets based on their relationship with agent a_i and stored in the environment model, denoted as $M_e = \{U_i, D_i, S_i, ...\}$, where

- $U_i : K \rightarrow 2^{\mathcal{A}}$: mapping from product types to upstream agents from which a_i can obtain the products
- $D_i : K \rightarrow 2^{\mathcal{A}}$: mapping from product types to downstream agents to which a_i can provide the products
- $S_i : M_c \rightarrow 2^{\mathcal{A}}$: mapping from a capability in agent a_i to the agents that have the same capability

- $T_i : K \to 2^{\mathcal{A}}$: mapping from a product type to the transportation agents that can reach agent a_i.

In this way, an agent is capable of identifying the subset of agents it needs to communicate with and exchange information in the network. The mapping functions U_i, D_i, and T_i identify the agents that can execute a physical product flow (e.g., obtaining products from agent a_i) with agent a_i. The information about the flow (e.g., product type and amount) is stored in the environment model via other mapping functions associated with the agent sets. For example, a function $U_i(K) \to K \times \mathbb{R}_+$ maps the upstream agent to the product type and amount of product flow with a_i. The mapping S_i identifies agents that perform the same behavior as agent a_i for replanning purposes.

The environment model is generated when the agent is initialized based on its associated physical entity. The agent sets and their associated information are dynamically updated if the supply chain environment changes, such as the production capabilities of a given agent change. Note that changes may occur during a time period t if the changes only occur in agent knowledge without leading to changes in the network.

State model The agent dynamics are described by a flow balance of varying input and output products. The state model describes the dynamics in terms of flow, production, and inventory based on the flow balance equation [17]:

$$I_{i,t+1} = I_{i,t} + u_{i,t} - z_{i,t} + h_{i,t}(I_{i,t}, u_{i,t}). \tag{3.6}$$

Note that each variable in the state model is a vector indexed by the product types that agent a_i needs or produces, where

- State vector $I_{i,t} = [I_{ik,t}]_{k \in K}$ represents the amounts of different products stored in the agent at time t.

- Input vector $u_{i,t} = \left[\sum_{j \in U_i(k)} y_{jik,t}\right]_{k \in K}$ represents the product flows coming from the upstream nodes.

- Output vector $z_{i,t} = \left[\sum_{j \in D_i(k)} y_{ijk,t}\right]_{k \in K}$ represents the product flows going to the downstream nodes.

- Production function $h_{i,t}(I_{i,t}, u_{i,t})$ gives the number of used components and produced products if the agent has production capability.

Note that the variables in (3.6) are bounded by agent-specific limits. For example, $I_{i,t}$ is limited by the inventory capacity of a_i, while $u_{i,t}$ and $z_{i,t}$ are limited by transportation capacities.

3.2.1.2 Desires

Desires represent the goals and requirements of an agent. In this work, the desires include the objective functions and the constraints for the decision-making of the agents. For example, an agent makes decisions to select suppliers with minimal cost (objective $\mathcal{J}$) considering the limit of the number of suppliers (constraint $\mathcal{K}$).

3.2.1.3 Intentions

Intentions represent the plans an agent has committed to executing. The intentions of an agent depend on its capabilities. For example, the intentions of a transportation agent include product flows it has committed to transporting, while for a supplier agent, the intentions describe the production and out-flow of products to downstream agents.

3.2.2 Decision and Communication Manager

3.2.2.1 Decision Manager

The Decision Manager of an agent consists of multiple decision-making models. Compared to manufacturing agents, the decisions that supply-chain agents make are different. In this section, we primarily discuss the decision-making for disruption identification, response optimization, and supplier selection.

Disruption Identifier Component that identifies the consequences of a disruption, such as the lost production and flow streams.

Response Optimizer Component that determines how an agent responds to other agent requests by solving an optimization model based on the response agent's objectives and constraints.

Supplier Selector Component that determines how an agent selects suppliers to satisfy its demand by solving an optimization model based on the selector agent's objectives and constraints.

3.2.2.2 Communication Manager

Provides the interface for the agent to exchange information with its physical entity and other cyber agents. In this work, components of the communication manager include an information manager, a request manager, and a response manager.

Information Manager Collects information from and sends decisions to the agent's associated physical entity.

Request Manager Sends requests from the Decision Manager to other agents and passes requests received from other agents to the Decision Manager.

Response Manager Sends responses from the Decision Manager to other agents and passes responses received from other agents to the Decision Manager.

3.3 Agent Coordination and Decision-Making

In this section, we describe the proposed agent communication strategy for disruption mitigation, including three processes: disruption identification, iterative communication for supplier reselection, and propagated communication. Figure 3.4 shows the overall flow of the agent decision-making and communication protocol for supplier selection. Algorithm 2 describes the detailed

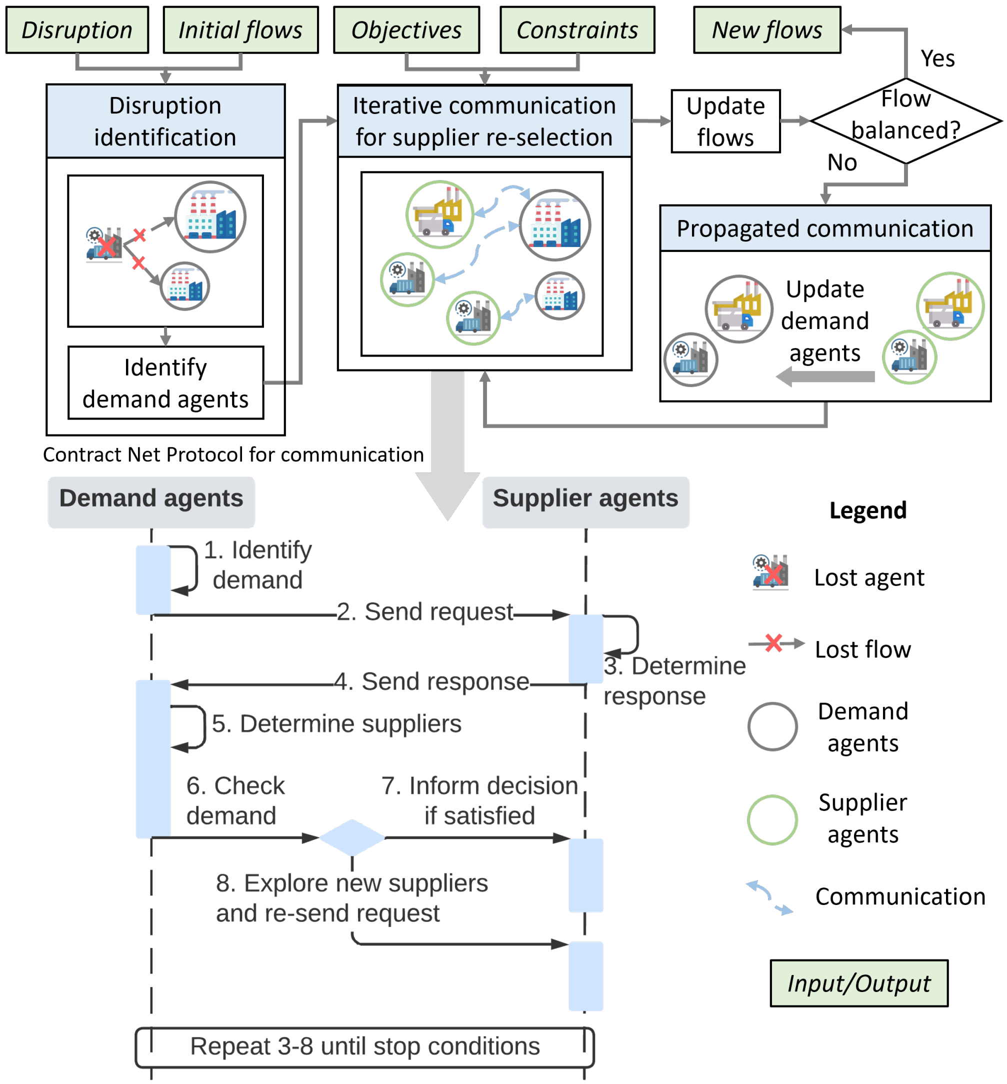

Figure 3.4: The flow chart of the proposed agent communication and decision-making process

overall decision-making processes while Algorithm 3 describes how agents iteratively communicate for supplier re-selection. We assume that agents can communicate regardless of the disruption and that communication links can be established or removed based on the decision-making strategy. Agent communication follows Contract Net Protocol (CNP) [110], which aligns with the Foundation for Intelligent Physical Agents (FIPA) standards [111].

Algorithm 2 Distributed decision-making for disruption mitigation

Input: $a_e, f_0, \mathcal{K}, \mathcal{J}$
Output: f_r
// Disruption and demand identification
1: $y_e^0 \leftarrow (y_{ejk}^0, y_{iek}^0)_{i,j \in V, k \in K} \subset f_0$ // Identify lost flows
2: $\mathcal{A}_{dm} \leftarrow \{a_j, \forall j \in y_{ejk}^0 \in y_e^0\}$ // Identify demand agents
3: $f_r \leftarrow f_0 \setminus y_e^0$ // Get remaining flows
// Agent communication
4: **while** $\mathcal{A}_{dm} \neq \emptyset$ **do**
  // Generate and update new product flows
5:   $\Delta f_r \leftarrow Algorithm\ 3$
6:   $f_r \leftarrow f_r \cup \Delta f_r$
  // Check flow balance and identify new demand agents
7:   **for** $y'_{zjk} \in \Delta f_r$ **do**
8:     **if** a_z needs materials/components **then**
9:       Add a_z to $\mathcal{A}_{dm}$
10:     **end if**
11:   **end for**
12: **end while**
13: **return** f_r

3.3.1 Disruption Identification

We focus on the disruption of an agent loss, which leads to losses of production and/or transportation flows. By periodically obtaining data and information from the physical entities, agents are able to detect disruptions that occur in their associated physical entities. Once a disruption occurs, the disrupted agent, denoted by a_e, checks its knowledge base to identify the lost flow y_e^0 related to it, as described in Algorithm 2, lines 1-3. Based on the lost flow, a_e initiates communication with downstream agents, defined as demand agents $\mathcal{A}_{dm}$, to inform them of the disruption to their incoming production flow streams. In addition, the a_e will inform upstream agents that provide products to the disrupted agent that the flow streams will be disrupted and may be reassigned to alternative agents. Since the flow streams f_r are no longer balanced and meeting the requested need, the demand agents $\mathcal{A}_{dm}$ must now find supplier agents that are capable of supplementing for the lost production of a_e.

Algorithm 3 Supplier re-selection via agent communication

Input: $\mathcal{A}_{dm}, y_e^0, \mathcal{K}, \mathcal{J}$
Output: Δf_r
// **Objective:** Find product flows to satisfy demand agents
while $\mathcal{A}_{dm} \neq \emptyset$ **do**
  // Request (all the demand agents)
  for $a_j \in A_{dm}$ **do**
    // Identify the need of each demand agent
    $d_{jk} \leftarrow y_{ejk}^0, \forall k \in K$
    // Identify agents to request
    a_j explores environments
    $\mathcal{M}_j(k) \leftarrow$ Environment model, $\forall d_{jk}$
    a_j requests $\mathcal{M}_j(k)$ for product need d_{jk}
  end for
  // Response (all the agents being requested)
  for $a_z \in \cup_{a_j \in A_{dm}} \mathcal{M}_j(k)$ **do**
    $\bar{y}_{zjk}, \hat{y}_{z*k} \leftarrow \min \mathcal{J}_z$
    a_z sends response $\bar{y}_{zjk}$ to a_j
  end for
  // Determine product flows and check need
  for $a_j \in \mathcal{A}_{dm}$ **do**
    $\hat{y}_{zjk} \leftarrow \min \mathcal{J}_j$
    $d_{jk} \leftarrow d_{jk} - \sum_{a_z} \hat{y}_{zjk}$
    if $d_{jk} < \epsilon$ **then**
      Δf_r appends $\hat{y}_{zjk}$
      $\mathcal{A}_{dm} \leftarrow \mathcal{A}_{dm} \setminus a_j$
    end if
  end for
end while
return Δf_r

3.3.2 Iterative Communication for Supplier Re-selection

Once the disrupted agent, a_e, identifies the lost flow streams and informs the downstream demand agents about this disruption, the demand agents must initiate communication with alternative suppliers. The contract net protocol (CNP) segment of the communication process illustrated in Figure 3.4 and Algorithm 3 describes the iterative communication process that consists of four key steps in each iteration: “Identifying needs”, “Requesting help”, “Responding to the requests”, and “Informing agents of accepted flow”. The outcome of this process is the selection of alternative suppliers to provide the necessary product flow that has been disrupted by the loss of an agent.

3.3.2.1 Identify needs

Based on the disrupted flow, a demand agent $a_j \in \mathcal{A}_{dm}$ identifies its current need, $d_{jk} = y^0_{ejk}, \forall k \in K$. Using its knowledge base, a demand agent then retrieves the set of objectives, $\mathcal{J}_{jk}$, and constraints, $\mathcal{K}_{jk}$, (e.g., budget, delivery date) to direct the decision-making process for a new supplier.

3.3.2.2 Request

Meanwhile, each demand agent a_j retrieves the environment models in its knowledge base and identifies the agents where it will send a request for additional flow. We define $\mathcal{M}_j(k)$ as a set of agents that a_j can request for product k. The demand agent a_j sends out requests, denoted by $Req = (d_{jk}, \mathcal{K}_{jk})$, to all agents in $\mathcal{M}_j(k), \forall k \in K$. Note that all of the demand agents may send requests in parallel.

3.3.2.3 Response

Request agents ($a_z \in \cup_{a_j \in \mathcal{A}_{dm}} \mathcal{M}_j(k)$), those that receive the request from the demand agents, will check their knowledge bases and determine their ability to provide the product flows (i.e., satisfy the requests) based on their objectives $\mathcal{J}_z$ and constraints $\mathcal{K}_z$. They will then formulate a response that includes available product flows, $\bar{y}_z = [\bar{y}_{zjk}, a_j \in \mathcal{A}_{dm}, k \in K]^\mathsf{T}$, where $\bar{y}_{zjk}$ represents the maximum units of product k that a_z can provide to a_j, and related information $\mathcal{F}_{zjk}$ (e.g., product cost). The request agents a_z determine their responses by solving a local (individual agent) optimization model. An example is given below:

$$\max_{\bar{y}_z} \quad \mathcal{J}_z(\bar{y}_z) = \sum_{a_j \in A_{dm}, k \in K} r_{zjk} \bar{y}_{zjk} \tag{3.7a}$$

$$\text{s.t.} \quad \sum_{k \in K} \bar{y}_{zjk} \leq q_{zj}, \ \forall a_j \in A_{dm} \tag{3.7b}$$

$$\sum_{a_j \in A_{dm}} \sum_{k \in K} \bar{y}_{zjk} \leq \bar{p}_z, \tag{3.7c}$$

$$\bar{y}_{zjk} \leq d_{jk}, \forall a_j \in A_{dm}, k \in K, \tag{3.7d}$$

where (3.7a) maximizes agent a_z's revenue, subject to the constraints of flow capacity (3.7b) and production capacity (3.7c) from the agent itself and the constraints of from the requested demands (3.7d). Note that agents can have their own specific objectives (e.g., inventory level and revenue) and constraints. All requested agents send their responses, i.e., $Res = (\bar{y}_{zjk}, \mathcal{F}_{zjk})$, back to the demand agents.

3.3.2.4 Inform

After receiving responses from all of the requested agents, each demand agent, a_j, determines the new product flow streams $\hat{y}_j = [\hat{y}_{zjk}, a_z \in Z_j(k), k \in K]^\mathsf{T}$ by solving another optimization. An example is shown below:

$$\min_{y'_j} \quad \mathcal{J}_j(y'_j, \mathcal{F}_{zjk}) = \sum_{a_z \in Z_j(k), k \in K} r_{zjk} y'_{zjk} + \sum_{k \in K} \rho^d_{jk} \Delta^d_{jk} \tag{3.8a}$$

$$\text{s.t.} \quad y'_{zjk} \leq \bar{y}_{zjk}, \ \forall z \in \mathcal{M}_j(k), k \in K, \tag{3.8b}$$

$$\Delta^d_{jk} \geq \sum_{a_z \in Z_j(k)} y'_{zjk} - d_{jk}, \ \forall k \in K \tag{3.8c}$$

$$\text{other agent constraints,} \tag{3.8d}$$

where (3.8a) minimizes the total costs and demand dissatisfaction of the demand agent. Constraint (3.8b) denotes that the chosen new product flows cannot exceed the suppliers' responses.

Constraint (3.8c) calculates the unmet demands. Constraint (3.8d) presents other constraints for supplier selection, such as a limited number of suppliers for the demand agent. Note that demand agents may have their own specific objectives and constraints. Then the demand agents check whether their decisions can satisfy their needs with an acceptable threshold. If so, the demand agents inform the chosen agents to provide new product flows.

3.3.2.5 Iterative communication

The above request-response-inform process takes $|A_{dm}| + |\cup_{a_j \in A_{dm}} \mathcal{M}_j(k)| + |A_{dm}|$ computations. However, if there are agents whose needs cannot be satisfied, they will explore the environment to identify other agents that can provide the needed products (i.e., identify new suppliers within $\mathcal{M}_j(k)$). These agents will then repeat the request-response-inform process to determine new product flows. The iteration process will stop once the demand agents have identified suppliers that can meet their needs or if it has been determined that there are no suitable agents capable of providing the needs. To guarantee convergence, we set an upper bound n_e for the number of times that demand agents can explore. Therefore, the complexity of Algorithm 3 is $\mathcal{O}(n_e \times \max\{|A_{dm}|, |\cup_{a_j \in A_{dm}} \mathcal{M}_j(k)|\})$. Note that $|A_{dm}|$ is related to the *Connectivity* of the disrupted agent and $|\cup_{a_j \in A_{dm}} \mathcal{M}_j(k)|$ is related to the *Connectivity* and *Complexity* of the disrupted agent. Therefore, if the disrupted agent has higher *Connectivity* and *Complexity*, the re-planning process generally requires more computations.

3.3.3 Communication Propagation

The iterative communication process occurs between the demand agents and their immediate supplier agents. However, this process may propagate through the entire network if the process of meeting the demands of certain agents introduces new needs from the suppliers meeting those demands. In this manner, the suppliers become new *demand* agents, resulting in a continuation of this process, as described in Algorithm 2, lines 4-12.

3.3.3.1 State update

Algorithm 3 identifies several new product flow streams that are necessary to satisfy the needs of the initial demand agents. Once these flow streams are identified, the demand agents and selected supplier agents must update their states, resulting in a change to the network flow states: $f_r \leftarrow f_r \cup \Delta f$. At this point, based on the objective of Algorithm 3, it is assumed that the demand agents have reached a balanced flow, while the selected supplier agents may need additional components in order to meet their new flow demands.

3.3.3.2 Propagation

Since each selected supplier agent, a_z, commits to providing products to meet the needs of the demand agents, this may introduce additional product/component needs from the suppliers to ensure sufficient products to meet these new commitments. In this case, the supplier agents no longer have balanced flow streams and must propagate demand requests in order to meet their needs to *their* related supplier agents. The communication process of Algorithm 3 will now be repeated; however, in this iteration, the selected supplier agents have become new demand agents. The propagation process stops when all of the agents have met their additional needs (e.g., the requests have been propagated through all upstream agents in the network). In this distributed decision-making strategy, the propagation process may vary from one level upstream through the entire network, depending on the production flow needs. Therefore, the number of times that Algorithm 3 is repeated depends on the *Depth* of the disrupted agent (D_d). The worst case is that the communication propagates to the most upstream agent, whose depth is $D_{max} = \max_{a_i \in V}\{D_i\}$. In this case, Algorithm 3 is repeated $\Delta D = D_{max} - D_d$ times. Therefore, the complexity of Algorithm 2 is $\mathcal{O}(\Delta D \times \mathcal{N}_{max})$, where $\mathcal{N}_{max}$ represents the maximum number of computations in the repeated Algorithm 3. Note that if the disrupted agent has higher *Depth*, the re-planning process requires less computations.

Overall, the proposed model-based agent knowledge provides heuristics to guide agent communication. In this way, agents are able to selectively communicate with other agents who possess

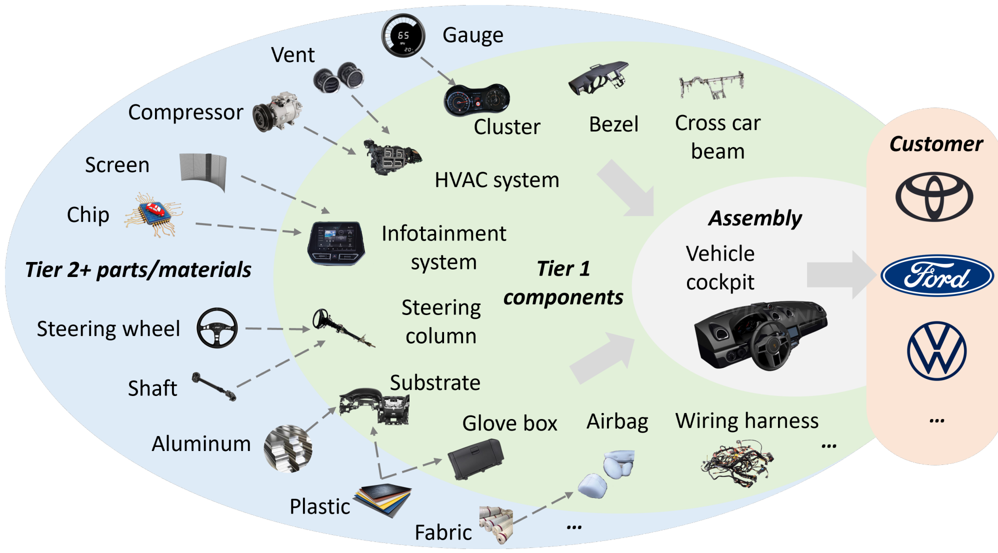

Figure 3.5: The simplified product structure for automotive cockpit

relevant abilities to handle the disruption, thereby reducing unnecessary communication. Moreover, the integration of agent exploration and iterative communication enables agents to thoroughly search for all potential solutions within the network. However, it is important to note that the optimality of this approach depends on how much the users allow agents to explore and communicate. There is a cost trade-off between exploration and hence increased communication requirements, and performance.

3.4 Case Studies

3.4.1 Case Study Set-up

To conduct numerical studies to evaluate the proposed approaches, we designed an example based on a supply chain for vehicle cockpits. In this section, we describe the supply chain instance, introduce several disruption scenarios, and derive metrics for performance evaluation.

3.4.1.1 Supply chain instance

Product structure We consider a vehicle cockpit supply chain, consisting of cockpit assembly plants, their customers (i.e., vehicle assembly plants), and their suppliers for components and materials. Here we summarize the supply chain product structure and network. Figure 3.6 shows that in our example vehicle cockpits represent the final product and are assembled using several manufactured components, comprised of different parts and/or materials.

In this instance, there are 3 different models of vehicles, and each model has demands for 1, 2, or 3 styles of cockpits. Between auto and cockpit assembly plants, we have the following assumptions:

- Each auto assembly plant only makes 1 type of vehicle model;
- Each cockpit assembly plant can produce multiple styles of cockpits.

Each cockpit requires 10 components to be assembled, as shown in the green ellipse in Figure 3.5, but different styles may need different component types. Between cockpit assembly and component suppliers, we have the following assumptions:

- The cockpits for the same auto model require the same type of cluster, substrate, glove box, HVAC system, cross-car beam, and steering column;
- Each style of cockpit requires a unique type of infotainment system, wiring harness, and a combination of different bezel types;
- All the cockpits use the same type of airbag.

Each component needs certain types and amounts of parts and/or materials in order to be produced. The blue ellipse in Figure 3.5 provides examples of some of the parts/materials needed for these components. Between component suppliers and part/material suppliers, we have the following assumptions:

- Different types of components may require different part/material types, yet they may share the same part/material suppliers;
- The part/material suppliers represent the most upstream suppliers in this instance.

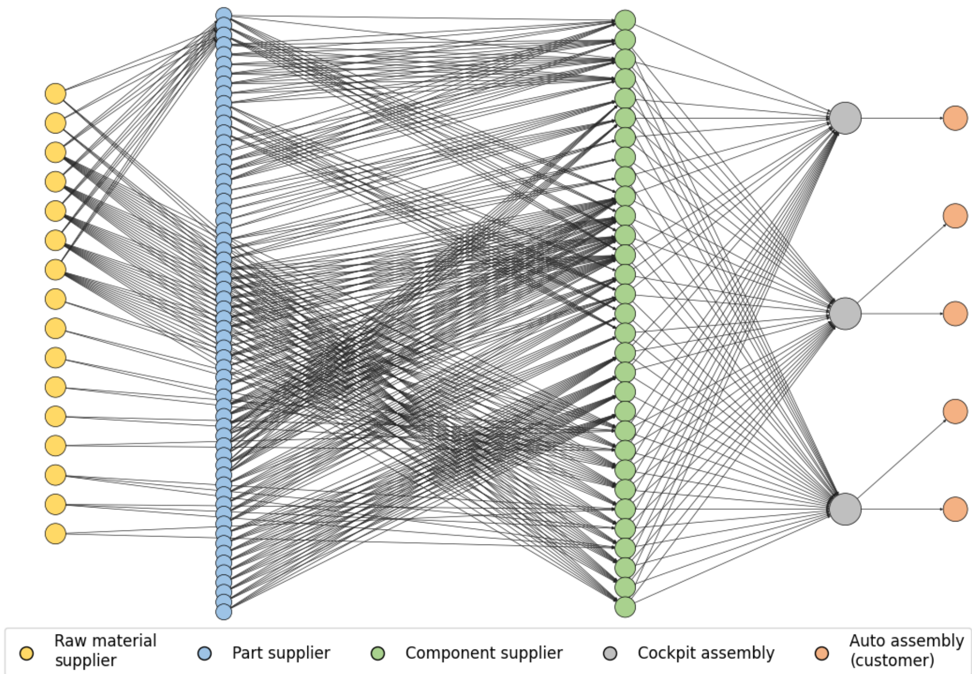

Figure 3.6: The supply chain network instance used for case studies.

Supply chain network Based on the product structure above, we designed a supply chain network with 117 supplier/customer agents and 413 transportation agents, as shown in Figure 3.6. These agents correspond to *physical* elements within the supply chain and are equipped with the proposed agent architecture to conduct communication and decision-making. The network instance contains 5 customers, 3 cockpit assembly plants, 31 component suppliers, 62 part suppliers, and 16 raw material suppliers that are connected via distinct transportation units. The 5 auto assembly plants represent customers that have placed demands for different style cockpits. Each cockpit assembly plant produces cockpits for a specific auto model type. For each type of component, part, and material, we have multiple suppliers that have production capabilities. Each supplier or transportation agent has its own production or transportation cost and capacity.

3.4.1.2 Benchmark and disruption scenarios

As discussed in Section 3.1, existing distributed approaches in the field of supply chain management typically rely either on pre-defined disruption scenarios or rule-based decision-making.

It is challenging to evaluate and compare these approaches effectively without access to their underlying agent design, database, and implementation details. Additionally, these methods do not focus on the problem of supply chain disruption response, thus comparing and assessing their performance accurately for this problem may be hindered, potentially leading to incomplete or misleading conclusions. Furthermore, though there are other distributed approaches utilized in other fields, such as manufacturing rescheduling, and multi-robot control, these approaches cannot be directly applied to the specific problems posed by supply chain disruption mitigation.

Therefore, to benchmark our distributed approach against a more common decision-making strategy, we evaluate the performance of both a centralized and distributed decision-making approach during various disruption scenarios. With visibility of all entities and their status in the supply chain network, the centralized model provides a highly communicative yet globally optimal reconfiguration plan. We use the centralized model that we developed in [17] to generate optimal initial product flows as a steady state before a disruption occurs. We apply the centralized model developed in [17] as a generic mixed-integer linear programming (MILP) model to optimize or re-optimize production, inventory, and transportation planning for a multi-echelon, multi-product supply chain network. Once a disruption is identified, the model is updated by modifying the network structures, parameters, and constraints. We then run the updated model to determine newly optimized decisions about the product flow and production schedule as a response to the disruption. Though it is not straightforward to analyze how the solver computes the solution for the centralized model, the input size of the centralized model is much larger than the proposed distributed approach. The centralized model re-optimizes all the agents considering associate constraints, while the proposed distributed approach only considers a subset of agents within the network.

The supply chain is initiated with a product flow plan derived by solving the centralized model optimization described above. The supply chain descriptions defined in Section 3.1.2.2 can be used to describe the role of each agent in this supply chain instance. Note that in the initial product flow plan, not all agents will have active production or transportation roles.

To evaluate a disruption from the loss of a single agent, we design the case study with the following rules:

- The lost agent should have production tasks in the initial plan;
- In each scenario, only one supplier agent becomes disrupted;
- Agents can exhibit production and transportation capabilities of 30% over their initially defined amount for an additional 50% unit cost.

In the pre-determined initial plan, there are 84 agents that exhibit production tasks, thus we run 84 scenarios, starting from upstream agents to downstream agents. For each scenario, we implement the centralized and distributed decision-making approaches from Section 3.3 to generate a new flow plan without the use of the lost agent.

The decision-making is focused on optimizing the system performance by minimizing a cost function, $\mathcal{J}$, at the network level (centralized) or local agent level (distributed):

$$
\begin{aligned}
\mathcal{J} = & \sum_{(i,j)\in E, k\in K} c_{ijk} y_{ijk} + \sum_{i\in V, k\in K} e_{ik} p_{ik} \\
& + \sum_{i\in V, k\in K} \rho^d_{ik} \Delta^d_{ik} + \sum_{(i,j)\in E} \rho^E_{ij} \Delta^E_{ij} + \sum_{i\in V} \rho^V_i \Delta^V_i,
\end{aligned} \tag{3.9}
$$

where the first two elements represent transportation and production costs when overcapacity must be applied, and the last three represent the penalty costs for unmet demand and the addition of new agents and edges. Note that at the local agent level, the cost function is applied across several local agents rather than the entire network as with the centralized approach. As mentioned in Section 3.3, an agent explores the environment to identify agents that can meet production needs in the case of a disruption. It is assumed that agents only interact with other agents that are within their network and therefore show up in their knowledge base. If the existing network cannot satisfy the required demands, an agent will seek to build connections with new agents. This exploration process will trigger a penalty if new agents are added to the existing network. The proposed cost function is used to provide an example. Additional objectives can also be investigated within this framework through the selection of different elements within the cost function.

3.4.1.3 Metrics for performance evaluation

In order to evaluate the impact of an agent's attributes within the network on the outcomes of different decision-making strategies, we define several key performance metrics.

Overage cost We define overage cost O as the total cost for any transportation or production flow that exceeds the original agent capacity. This metric represents additional efforts by the agents to address the disruption.

$$O = \sum_{(i,j)\in E, k\in K} \alpha_{ij} c_{ijk} y^o_{ijk} + \sum_{i\in V, k\in K} \beta_i e_{ik} p^o_{ik}, \tag{3.10}$$

where α_{ij} and β_i are the multipliers for the increased cost of over-capacity flow and production; y^o_{ijk} and p^o_{ik} are the amount of over-capacity flow and production that are determined by the optimization.

Network changes We define network changes N_c as the sum of the number of agents that changed their existing production amount plus the number of flow channels that are changed in terms of the type and/or amount of products.

$$N_c = |\{a_i | p_i \neq p'_i, \forall i \in V\}| + |\{(i,j) | y_{ij} \neq y'_{ij}, \forall (i,j) \in E\}|, \tag{3.11}$$

In (3.11), p_i is the initial production of a_i and y_{ij} is the initial flow of edge (i, j); p'_i and y'_{ij} are the new production and flow, respectively.

Network additions We define Network additions N_a as the sum of the number of new agents and edges that are added to the network to address the disruption.

$$N_a = \sum_{i\in V} \max\{0, \xi' - \xi\} + \sum_{(i,j)\in E} \max\{0, \zeta' - \zeta\} \tag{3.12}$$

Here ξ and ζ indicate the usage of agents and edges in the initial plan (1 if used, 0 otherwise), respectively, and ξ' and ζ' represent the new plan. Both N_c and N_a represent how the network changes to respond to the disruption in terms of the overall production and flow in the network. In practice, changing existing flow streams and production types or introducing new suppliers and transportation units may require a significant amount of business effort, and may not be practical in many instances.

Agent communication We define agent communication M as the number of communication exchanges used to determine a response to the disruption. The communication effort M in the centralized method includes the request for re-running the model, the requests to and responses from all the agents in the supply chain to collect information, and the notifications to the agents whose flow and/or production plan need to change:

$$M = 1 + 2|\mathcal{V}| + N_c + N_a \tag{3.13}$$

In the distributed method, M includes all the agent requests, responses, and inform messages, as defined in Section 3.3.

3.4.2 Case Studies: Re-planning for an agent loss

In this section, we present a summary of the case study results for the various scenarios and investigate how agent attributes, within the context of a specific decision-making approach, impact the different performance metrics. Based on these results, we provide insight for users to consider when determining an approach to use for disruption response.

3.4.2.1 Overview of the results

In this case study we evaluated 84 disruption scenarios. Within these 84 cases, there were 11 scenarios in which at least one of the approaches was unable to find a solution that could satisfy

Table 3.2: Metric evaluation across individual scenarios for supply chain re-planning case studies

Metrics	Number of scenarios (73 in total) where	
	distributed approach performs better	two approaches are similar
Network changes N_c	73	0
Communication M	73	0
Overage cost O	18	42
Network additions N_a	0	42

all of the customer demands. In these scenarios, the network exhibited a redundancy of zero, $\mathcal{R}_i(m) = 0$, or insufficient remaining capacity to recover all the production losses of the disrupted agent. For these cases, the centralized approach was used to meet the demand by re-optimizing the entire supply chain network to redistribute the capacity and production capabilities. For the remaining 73 scenarios, the distributed and centralized approaches were able to find new plans that satisfied all the demands. Importantly, the computation time for the distributed decision-making approach was 99% faster than the centralized approach. The obtained results in our study provide validation for our assumption that the redundancy of the disrupted agent is indeed a necessary condition for recovering performance in supply chain networks. The presence of redundant agents plays a crucial role in maintaining and restoring the overall system performance in the face of disruptions. Moreover, the proposed distributed approach demonstrates its capability to find a solution for the loss of any arbitrary agent as long as a viable solution exists within the network.

A comparison of the performance of the supply chain reconfiguration as a function of the decision strategy and performance metrics is shown in Table 3.2. Network changes and communication use are minimized by the distributed approach, while overage costs and network additions generally result in similar performances for either the centralized or distributed approach. In the following analysis, we focus on the 73 scenarios where the production demands are satisfied.

3.4.2.2 Performance

In this section, we investigate how agent attributes impact the performance metrics for both centralized and distributed approaches. Note that since redundancy mainly affects demand satis-

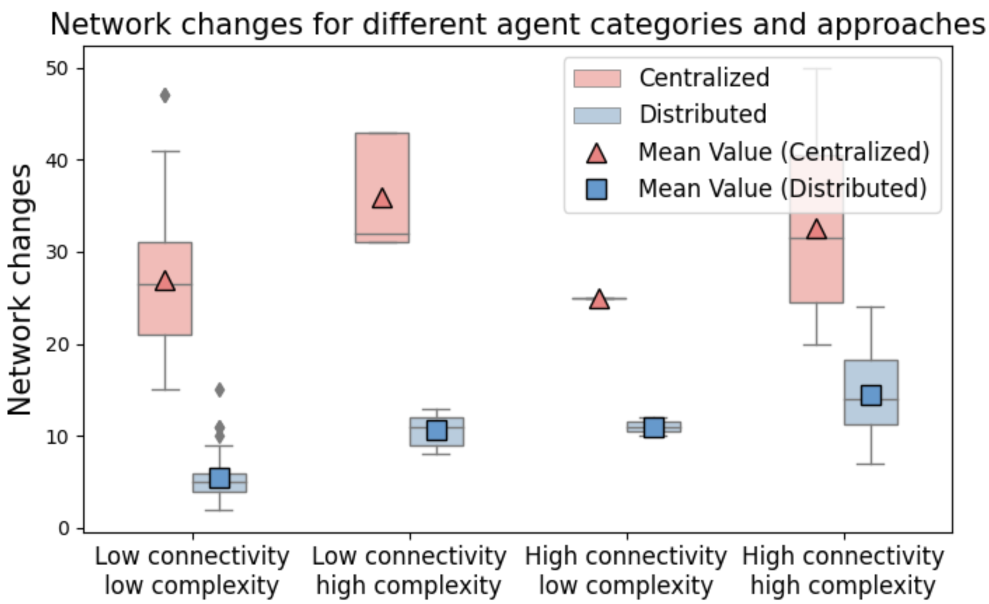

Figure 3.7: Number of network changes for different categorized agents based on attributes using centralized and distributed approaches.

faction and depth only affects partial metrics, we focus on agent connectivity and complexity. To illustrate how the attributes affect performance, we categorize the 73 agents into four categories, which are combinations of low and high connectivity and complexity. The connectivity of the 73 agents goes from 1 to 11, and we choose 5 as the cutoff between low and high. The complexity of the 73 agents goes from 1 to 15, and we choose 7 as the cutoff between low and high. Based on these criteria, there are 56 agents with low connectivity and low complexity, 5 agents with low connectivity and high complexity, 2 agents with high connectivity and low complexity, and 10 agents with high connectivity and high complexity.

Network changes As shown in Table 3.2, the distributed approach results in fewer network changes than the centralized approach in all 73 scenarios, while the exact number of network changes is related to the attributes of the lost agent. We categorize agents based on their connectivity and complexity, and present the outcome of these categorized scenarios, as shown in Figure 3.7. For the centralized approach, agent connectivity has minimal impact on network changes; however, if the lost agent has high production complexity, the centralized approach causes more network changes than the scenarios where the lost agent has low complexity. Since the centralized approach minimizes the total objective without considering how it will change the production and transportation of individual agents, the results show that the capability attribute (i.e., complexity) has more effect on network changes in the centralized approach than topological attributes (i.e., connectivity).

For the distributed approach, we observe that agents with high connectivity in the network (topology perspective) or high product complexity (capability perspective) result in more changes to the network. Specifically, Figure 3.7 shows that the number of network changes increases as either connectivity or complexity become higher. These two attributes have a similar effect on the network changes for the distributed approach since they both determine whether the agent needs to propagate its local negotiation to other agents, thus leading to additional network changes.

In addition, the difference in network changes between the centralized and distributed approaches becomes smaller when the disrupted agent has high connectivity and complexity. This is because these agents may require communication across a large portion of the network, leading to additional network changes that mirror the quantification of changes from the centralized approach.

Summary: High complexity leads to more network changes for both the centralized and distributed approaches, while high connectivity only impacts the distributed approach.

Communication As shown in Table 3.2, the distributed approach requires less communication than the centralized approach in all 73 scenarios. Figure 3.7 shows how agent connectivity and complexity impact communication. For the centralized approach, communication is not influenced by the level of connectivity and complexity of the disrupted agent. As defined by (3.13), the communication for the centralized approach is dependent on the network size $|\mathcal{V}|$.

For the distributed approach, higher connectivity or complexity leads to more agent communication. Similar to the performance in network changes, communication showcases the distributed approach to computing a new plan using local negotiation and only propagates communication as

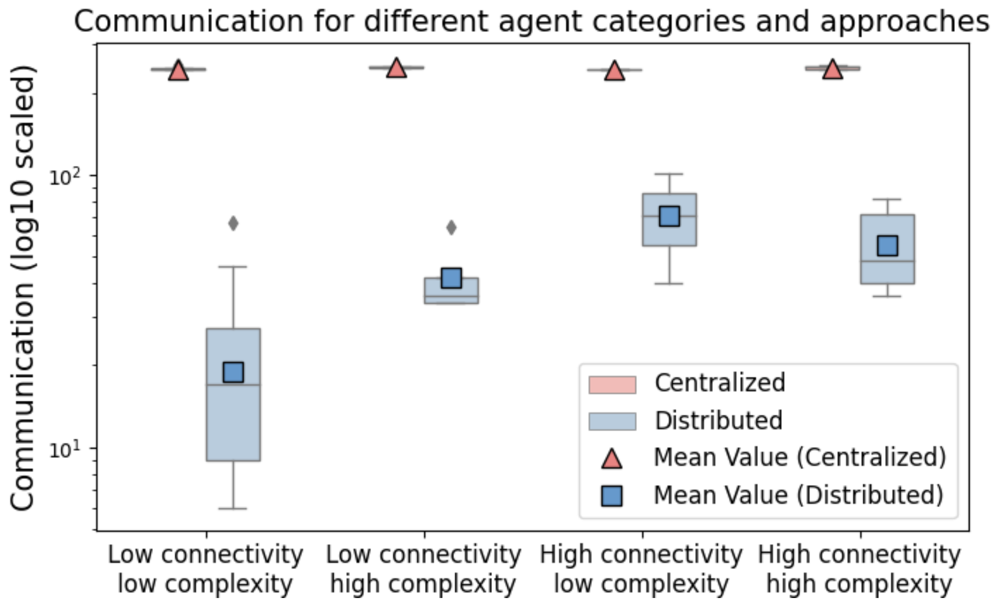

Figure 3.8: Number of agent communication exchanges for different categorized agents based on attributes using centralized and distributed approaches.

needed. However, for the agents with high connectivity, the level of complexity does not impact the communication needs for the distributed approach. This indicates that the topological attribute connectivity has more impact on communication than production complexity since it reflects the ripple effect that may go through the supply chain and lead to more communication.

The difference in communication between the centralized and distributed approaches also decreases for agents with high connectivity or complexity, since these agents may require communication through the entire network.

Summary: Connectivity and complexity do not impact communication requirements for the centralized approach. High connectivity and complexity lead to more communication for the distributed approach.

Overage cost One might expect that a centralized approach, which optimizes flow across the entire supply chain network, would result in lower production costs, especially when overage costs are taken into consideration. However, Table 3.2 shows that out of the total 73 scenarios, there were 18 scenarios where the distributed approach computed lower-cost solutions, and 42 scenarios

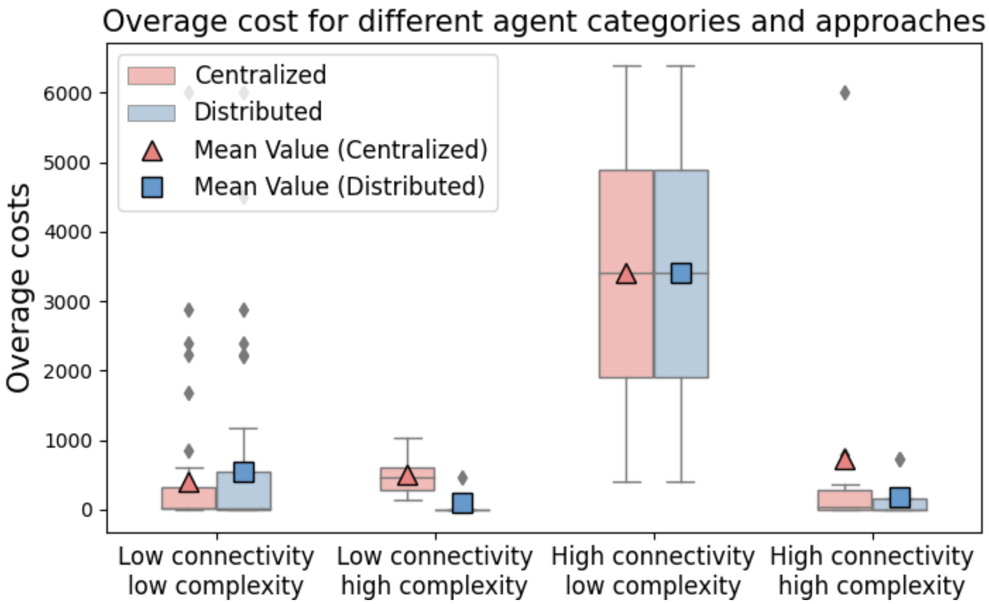

Figure 3.9: Overage cost for different categorized agents based on attributes using centralized and distributed approaches.

where the distributed approach provided plans with similar costs to the centralized approach.

To investigate these results, we present the overage costs based on different agent attributes, as shown in Figure 3.9. For both the centralized and distributed approaches, high overage costs come from low-complexity agents. Low complexity represents agents that require a small number of components for production or that produce variants that have limited use in the final products. Such an agent generally has a limited number of redundant suppliers with limited excess capacity. This leads to high overage costs in order to meet production needs. Overage is related to capability rather than topological attributes.

The influence of the overage cost differs depending on the decision-making strategy. The distributed approach does not provide a global view, thus this approach selects several backup suppliers so that the over-capacity is low, which causes more costs through added edges. The centralized approach has a full network-level view, and it chooses one backup supplier with over-capacity to avoid adding additional flow channels since the centralized objective contains a large penalty for adding new agents and edges.

Summary: Low complexity leads to higher overage costs for both centralized and distributed

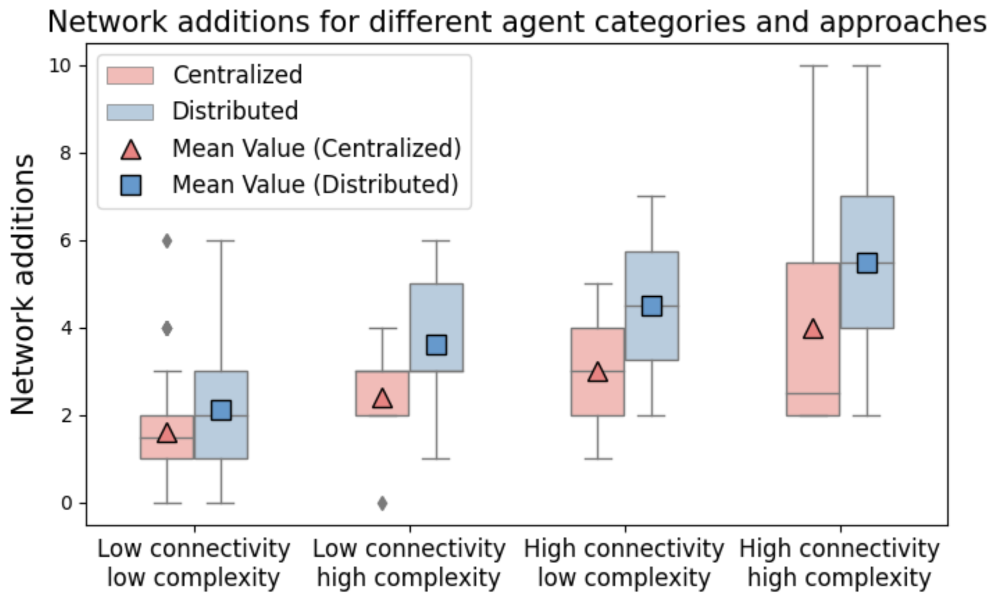

Figure 3.10: The numbers of network additions for different categorized agents based on attributes using centralized and distributed approaches.

approaches while connectivity has no effect.

Network additions As shown in Table 3.2, the distributed approach results in similar network additions in 42 of 73 scenarios, while the centralized approach provides fewer network additions in the other scenarios. Figure 3.7 shows that for both the centralized and distributed approaches, the network additions increase as connectivity and complexity increase. Though the connectivity and complexity seem not to impact network additions significantly, it could be limited by the instance since there are not lots of unused agents to be added as a disruption response.

In addition, the distributed approach results in more network additions than the centralized approach when the lost agents have high complexity. As discussed above for the overage cost, the distributed approach selects several backup suppliers to lower over-capacity productions and flows but causes more added edges due to its local view of the network. The centralized approach utilizes its network-level view to minimize the overall additional agents and edges.

Summary: High connectivity or high complexity can result in more network additions for both centralized and distributed approaches.

3.4.3 Managerial Insights

From the results above, we can derive some insight into how agent attributes impact the performance of disruption response for centralized and distributed approaches.

3.4.3.1 Agent attributes

The results show that the centralized approach's performance is more affected by agent complexity (capability attribute) than agent connectivity (topological attribute). This conclusion validates the study in [103], which states that high-connectivity agents are not necessarily critical to disruptions. For the distributed approach, both complexity and connectivity impact the performance metrics mentioned above. It can be concluded that the performance of the distributed approach appears to be more sensitive to agent attributes than the centralized approach. Therefore, additional agent attributes should be investigated to further analyze the performance of the distributed approach. The conclusions from this work indicate, to some degree, that capability attributes are important to be considered in supply chain models, no matter what decision-making approaches to be applied.

3.4.3.2 Performance evaluations

From the performance perspective, the results above provide information for supply chain managers about how agent connectivity and complexity impact a specific set of performance metrics for both centralized and distributed approaches. However, in practice, enterprises and practitioners usually aim for multiple objectives. Though users could define any objectives to be optimized, numerical issues may occur if there are multiple objectives, e.g., hard to determine different weights. For this case study, in the vast majority of scenarios, the distributed approach provides solutions that have similar objective value to the centralized approach while also requiring fewer network changes and communication. This indicates the distributed approach may provide faster solutions that do not rely on information from the entire network at the cost of overage expenses and local optimality. The theoretical analysis presented in Section 3.3.2 and description of the centralized model in Section 3.4.1 demonstrate the potential reduction in both communication and computational efforts achieved by our distributed approach. By leveraging local communication and the model-based agent knowledge, the proposed framework reduces extensive information exchange across the entire supply chain network, leading to more efficient decision-making. However, the optimality of the distributed approach is contingent upon user-defined objectives and allows agent exploration and iterative communications.

3.4.3.3 Decision-making approaches

The proposed distributed approach can serve as an alternative strategy in situations where centralized approaches face challenges, such as agile response requirements and high heterogeneity within the supply chain network. The individual design of agents provides flexibility to manage the supply chain heterogeneously, and agents' local communication enables quick responses. However, unlike centralized approaches, the local view of agents using distributed approach may result in the potential of losing optimality. Therefore, it is important to understand how different agent attributes impact the effectiveness and performance of both centralized and distributed approaches. One example in the results above is that when the disrupted agent has high connectivity and complexity, the distributed approach tends towards that of the centralized approach. In this scenario, a large amount of communication has to be used to determine a response to the disrupted agent. In practice, the choice of the decision-making approach largely depends on the time scale of the disruption. If an agent is expected to be offline for a short time, the distributed approach can give a good solution quickly, with minimal changes to the rest of the supply chain. For a long-term disruption, it may be worthwhile to re-run the centralized model to provide a new globally optimal plan, although a short-term modification based on the distributed approach may provide a good temporary solution. To conclude, this work can be used to provide valuable insights for decision-makers to choose strategies depending on disruptions to enhance supply chain resilience and achieve better overall performance.

3.4.3.4 Generality

Based on the complexity analysis and the tested 84 agents in the case study, these derived insights can be generalized to some extent to other supply chains. However, the insights should not be interpreted as definitive inference statements. For example, one cannot conclude that agents with higher connectivity will always result in more network changes compared to agents with lower connectivity. More importantly, this work presents a generalized multi-agent framework that allows for investigating the correlations between agent attributes and performance. Users have the flexibility to model their own supply chains, customize metrics, and test different disruption scenarios. It is important to highlight that although our focus is on the disruption caused by the loss of an agent, this proposed approach is also applicable to disruptions related to new customer demand. In such cases, the customer itself becomes the disrupted agent as well as the demand agent, triggering the proposed agent communication strategy. However, for other types of disruptions, such as lead time disruptions [112] or the introduction of new agents into the network, the current framework would need to be extended to incorporate these features. Further research and development would be necessary to expand the capabilities of the framework to handle such disruptions effectively.

3.5 Concluding Remarks

In this chapter, a model-based multi-agent framework is developed and applied to supply chain networks. The agent architecture follows the same design principles as Chapter 2 but with different models for the Knowledge Base and Decision Manager of agents. Taking agent knowledge as heuristics, this chapter develops a flexible and agile iterative communication strategy to recover a new product flow plan without requiring prior knowledge of the potential disruptions. The proposed framework can be used to create supply chain models that enable dynamic and agile responses to different disruptions (e.g., supplier loss and new customer demand) and allow for greater agent flexibility and scalability to larger systems when compared to rule-based architectures.

In addition, we provide supply chain descriptions from a capability and topology perspective and describe individual enterprise agents as a function of these network attributes. Based on these descriptions, we investigate the impact of network attributes on the decision-making strategy used to address supply chain disruptions. To conduct this performance comparison, we apply a standard centralized modeling approach and our proposed distributed agent-based approach to a disrupted complex supply chain network. Through the case study, we showcase the feasibility of the proposed multi-agent framework and analyze the performance of the decision-making strategies with several metrics based on the attributes of the disrupted supply chain entities in a complex supply chain network. The results illustrate the distributed approach could be used to determine a response for small-scale disruptions, as well as provide a temporary solution for large-scale disruptions. This framework can also be used to provide information as a decision support system to determine a decision-making strategy that optimizes user-defined performance metrics in response to supply chain disruptions.

However, though providing flexibility and agility, this chapter does not guarantee the resiliency of the new supply chain plan since it does not include uncertainties and risks in agent communication and decision-making. Therefore, in the next chapter, we investigate how uncertainties and risks can be modeled and incorporated into the proposed distributed approach.

CHAPTER 4

Heterogeneous Risk Management in Supply Chain Networks

This chapter presents a heterogeneous risk management mechanism to incorporate uncertainties and risks into agent communication and decision-making, aiming for a resilient disruption response in the supply chain network toward maintaining performance against uncertainties. As described in Chapter 3, focusing on deterministic re-planning has limited resiliency since real-world industrial environments are stochastic. In the existing literature, most agent-based disruption response strategies focus on identifying risk mitigation actions from the system level without considering individual agent risk management [73, 74], and they do not consider the uncertainties in these actions to provide a resilient solution. These approaches have limited resiliency and also neglect the natural heterogeneity of supply chain networks. In this chapter, we focus on resilient decision-making for disruption responses in supply chain networks. Following the communication strategy presented in Chapter 3, we incorporate a risk-aware stochastic optimization to improve resiliency during a disruption response.

The rest of the chapter is organized as follows. A literature review and problem formulation are presented in Section 4.1. Section 4.2 presents the proposed heterogeneous risk management mechanism and describes how it is applied to supply chain disruption response problems. Section 4.3 discusses a case study that demonstrates the ability of the proposed framework to consider heterogeneous risk attitudes within the decision-making strategy. Concluding remarks are presented in Section 4.4.

4.1 Background and Overview

4.1.1 Literature Review

As discussed in Chapter 3, in the supply chain domain, most research focuses on centralized risk management methods to provide optimal solutions based on specific objectives (e.g., product flow cost) [20, 71, 72]. These approaches consider holistic risk at the supply chain level in a centralized decision model. However, modern supply chains are heterogeneous, where agents in the supply chain network play different roles and may possess diverse objectives [32]. Therefore, for risk management, it is important to distinguish the agents' risk attitudes, which may change dynamically as the supply chain environment evolves. Including this heterogeneity using a single overall model could result in a complicated objective function and many constraints since each agent has to be specified. There is a need for a distributed approach that is capable of addressing this heterogeneity, especially when rapid adaption is required.

Multi-agent systems enable heterogeneity and can improve the agility of supply chain risk management [32, 46, 47]. Agents make their own decisions based on their goals and knowledge and information obtained from other agents [7, 46]. Therefore, agents can choose different risk-based models to solve local problems, depending on the information available to them, their different risk attitudes, and their goals. Such flexibility allows each agent to evaluate risk differently, which is difficult and complicated to achieve in a centralized formulation, especially for large-scale and complex supply chains.

In the existing literature, most agent-based disruption response strategies use case-based and/or rule-based decision-making [37, 57, 68], and thus require prior knowledge of disruptions and reactive actions. These approaches focus on identifying risk mitigation actions from the system level without considering individual agent risk management and tolerance, and they do not consider the uncertainty in these actions to provide a resilient solution. Although some researchers derive individual agent risk models, they do not focus on the disruption response problem. For example, in [73, 74], all of the agents have identical risk models and conduct decision-making to improve supply chain design and address inaccurate demand prediction. In [75–78], agents have identical risk attitudes with different parameters showing how much the models consider the risk. The work in [113] allows different agent risk attitudes but only for customer agents. None of these approaches make full use of the heterogeneity of multi-agent systems to support heterogeneous and dynamic risk management.

To the best of our knowledge, no study has proposed a distributed decision-making approach that supports dynamic and heterogeneous risk management to provide a resilient response to disruptions in supply chains. To address the gap, the main contributions of this chapter are: (1) the development of a heterogeneous and dynamic risk management mechanism for supply-chain agents, (2) the incorporation of risk-aware stochastic optimization for resilient decision-making in response to supply-chain disruptions, and (3) an evaluation of the resiliency of the proposed approach through a simulated case study.

4.1.2 Problem Overview and Formulation

In this chapter, we focus on the following problem: given a supply chain network, individual enterprise agents, existing product flows, and a stochastic disruptive event at a supply-chain agent, how can we model and incorporate agents' risk attitudes and preferences into the decision-making to improve the resiliency of the disruption response? We make the following assumptions to specify our scope:

- A.1 The given supply chain starts from an original plan that is pre-determined.
- A.2 An unexpected disruption increases an agent's lead time and delays delivery. The associated agent can detect this disruption.
- A.3 The only uncertain parameters are production and lead time with known probability distributions inferred from historical data.

A.1 indicates that the supply chain follows an original flow plan before the disruption occurs. We describe the plan as all of the product flows (y_{ijk}) and arrival times (v_{ijk}) from agent a_i to a_j

for product k in the network. A.2 guarantees that disruption will be identified by the agent when it occurs and also designates how the supply chain network will be impacted by the disruption. A.3 defines the uncertainties in the supply chain and enables the quantification of risk attitudes that are incorporated into agent decision-making.

The disruption will trigger agents to re-organize the flow plan to minimize the production and flow costs, as well as the penalties for the risk of demand dissatisfaction regarding product amount and delivery time due to the disruption in lead time and delays in the delivery.

$$\min_{\hat{y},\hat{v}} \text{Cost} + \mathcal{H}_p(\hat{y},\hat{v}) + \mathcal{H}_t(\hat{y},\hat{v}) \tag{4.1a}$$

$$\text{s.t.} \quad \text{Agent constraints} \tag{4.1b}$$

$$\text{Network constraints} \tag{4.1c}$$

where $\hat{y}$ and $\hat{v}$ represent the new flows and arrival times, and $\mathcal{H}_p(\hat{y},\hat{v})$ and $\mathcal{H}_t(\hat{y},\hat{v})$ compute the penalties for the risks of unmet demand and delivery lateness for all customers, respectively. Note that these penalties arise from the uncertainties inherent in the constraints, making the optimization problem (4.1) a stochastic model. Some examples of the constraints can be found in Chapter 3.

Instead of resolving the centralized model, we apply the agent-based distributed approach that is proposed in Chapter 3 to provide a new flow plan. We revise the agent optimization by incorporating the uncertainties of production capacity and lead time. In this case, agents have different ways of handling uncertainties in the constraints and calculating the objective based on their own risk attitudes. Details are discussed in Section 4.2.2.

4.2 Heterogeneous Risk Management

4.2.1 Risk Heterogeneity in Multi-agent Supply Chain

In this section, we introduce a heterogeneous risk management mechanism for supply chain networks to guide the communication and decision-making for a disruption event. Each agent,

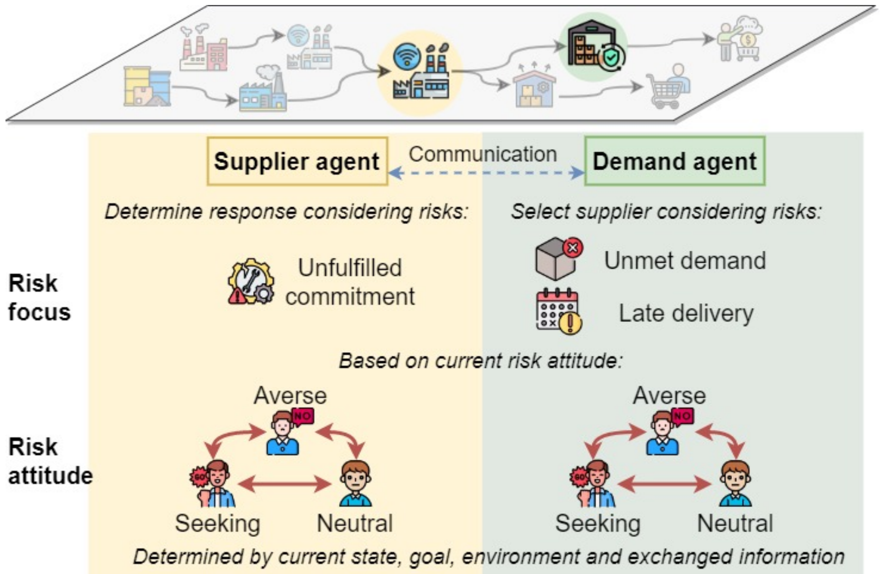

Figure 4.1: Risk management mechanism with heterogeneous risk focuses and risk attitudes for the roles of supplier agents and demand agents.

as an individual entity, considers their own risks and applies a different risk attitude depending on their role in the supply chain network and their current status. We define two types of risk heterogeneity in the supply chain network: agent risk focus and agent risk attitudes, as shown in Fig. 4.1.

4.2.1.1 Heterogeneity of agent risk focus

Based on the agent communication strategy for the re-planning problem in Chapter 3, we define two roles, supplier agents and demand agents. The supplier agents receive demand requests and provide products if they are selected. The demand agents need a certain amount of products at a given time to guarantee their scheduled production plans. Every agent in the supply chain network can be both a supplier agent and a demand agent in different scenarios. Therefore, agents consider different risks when they play different roles, which results in a dynamic and typically heterogeneous risk focus across the network.

Risks for supplier agents As mentioned in Chapter 3, when a supplier agent a_z receives demand requests from multiple demand agents, it makes decisions on how it responds to these requests depending on an evaluation of its capabilities. The objective of decision-making optimization is to seek maximal income and rewards while taking the risk of failing to fulfill a commitment and the penalty associated with the failure to fulfill a commitment into consideration. Each supplier agent maximizes the following function (additional details are included in Section 4.2.2):

$$\mathcal{J}_s = \sum_{a_j \in \mathcal{A}_{dm}, k \in K} r_{zjk} \bar{y}_{zjk} + w_s^e \sum_{a_j \in \mathcal{A}_{dm}} g_j \eta_j - w_s^r R_s \tag{4.2}$$

where r_{zjk} represents the income per unit that the supplier can earn from demand agent a_j for product k and $\bar{y}_{zjk}$ is the decision variable representing the quantity that supplier a_z plans to commit to demand agent a_j for product k. Parameter g_j indicates the rewards that supplier agents can gain from demand agent a_j if the product fulfillment is satisfied ($\eta_j = 1$). The rewards could include a bonus or future contract if the supplier agents can satisfy all the demands. R_s evaluates the risk of not fulfilling the response due to production and lead time uncertainties. The weights w_s^e and w_s^r are used to weigh the importance of rewards and risks. Note that different supplier agents could have different weights or other objectives and risks to consider. Supplier agents should consider the risk of unfulfilled commitment when they make decisions on their response and balance the trade-off between risk and demand fulfillment. The details of the optimization model are in Section 4.2.2.2.

Risks for demand agents The demand agents make supplier selection decisions based on the response they get from supplier agents. The objective of the supplier selection is to identify suppliers that can provide the required products while minimizing the costs associated with unmet demand and excessive time delays. A cost function describing this is shown below:

$$\mathcal{J}_d = C_d + w_d^t \sum_{k \in K} \Delta_{jk}^t + w_d^p \sum_{k \in K} \Delta_{jk}^p \tag{4.3}$$

where C_d represents the cost of obtaining products from supplier agents; Δ_{jk}^t represents the delay times, and Δ_{jk}^p represents the amount of unmet demand of product k; the weights w_d^t and w_d^p are used to weight the importance of unmet demand versus delay time for the specific demand agent.

Different demand agents can have different weights or alternative objectives and risks to consider. Depending on the uncertainty associated with a specific supplier's production capacity or lead time, demand agents will evaluate the associated risks and make a selection decision depending on their perceived risk attitude. The details of the optimization model are in Section 4.2.2.3.

4.2.1.2 Heterogeneity of agent risk attitude

In addition to the risk focus, agents may have different risk attitudes, which represent how agents balance risks and their original performance objectives (e.g., cost, revenue) depending on their current status. We consider states $X_r = \{averse, neutral\}$ to represent the different risk attitudes that agents could be. Specifically, an *averse* state indicates that agents try to make conservative decisions, i.e., avoid deviations between their behaviors and decisions. For agents in a *neutral* state, their decision-making aims to balance their objectives and risk assessment and avoid both conservative and risky decisions. Therefore, risk attitudes correspond to how agents measure the consequences of uncertainty, i.e., how they take into account risk and whether agents are risk-averse or risk-neutral.

In addition, agent risk attitudes may change dynamically as agents communicate and make decisions. For example, an agent could be risk-neutral when it supplies products but risk-averse when it demands products. Also, a supplier agent could be risk-neutral when it has an optimistic estimation of its production but risk-averse if not. Therefore, using this approach, this multi-agent supply chain network allows heterogeneous and dynamically-changing risk focus, attitudes, and tolerance for each agent. Through this proposed risk management mechanism, agents are able to incorporate risk assessment into their decision-making to handle uncertainties. By balancing the original objectives and risks, agents can make resilient decisions to respond to different disruptions.

4.2.2 Agent Decision-Making with Risks

In this section, we apply the proposed heterogeneous risk management mechanism to the same agent communication and decision-making strategy proposed in Chapter 3. Since this work focuses

Table 4.1: Nomenclature for agent decision-making.

Known Parameters	
d_{jk}	demand of product k at agent a_j
t_{jk}	deadline for agent a_j to receive product k
v'_{ejk}	new arrival time of product k flowing from agent a_e to agent a_j after disruption occurs
$\mathcal{A}_{dm}$	set of demand agents
$Z_j(k)$	set of supplier agents that can provide product k to agent a_j
r_{zjk}	unit income that agent a_z receives from a_j for product k
g^p_{zj}, g^t_{zj}	rewards that agent a_j provides to a_z if the response from a_z can meet the delivery time and demand
$\tilde{\ell}_{ejk}$	lead time of product k flowing from agent a_e to agent a_j
$\tilde{p}_{zk}$	estimated quantity of product k that agent a_z can produce
$\tilde{o}_{zk}$	estimated time that a_z can start to produce product k
Decision Variables	
$\bar{y}^u_{zjk}, \bar{y}^o_{zjk}$	maximum supply of product k that agent a_z responds to a_j within and over its production estimations
$\bar{v}^u_{zjk}, \bar{v}^o_{zjk}$	arrival times of $\bar{y}^u_{zjk}$ and $\bar{y}^o_{zjk}$ from a_z to a_j
$\eta^p_{zjk}, \eta^t_{zjk}$	binary variables that indicate whether the response from a_z can satisfy the demand and delivery time of a_j for product k
$\gamma^u_{zjk}, \gamma^o_{zjk}$	binary variables that indicate whether a_z decides to respond $\bar{y}^u_{zjk}$ and $\bar{y}^o_{zjk}$ to a_j for product k
$\Delta^p_{jk}, \Delta^t_{jk}$	units of unmet demand and delivery lateness of product k for the supplier selection of agent a_j
$\hat{y}_{zjk}$	units of product k that agent a_j agrees to obtain from a_z
$\lambda^u_{zjk}, \lambda^o_{zjk}$	binary variables that indicate whether a_j decides to select response from a_z for $\bar{y}^u_{zjk}$ and $\bar{y}^o_{zjk}$

on introducing uncertainties and risks into agent decision-making, we provide a single iteration of agent communication as an example. All the notations used in agent decision-making are summarized in Table 4.1.

4.2.2.1 Disruption identification

In this example, we aim to handle the disruption that increases an agent's lead time and results in late delivery to downstream agents. We define y_{ejk} as the quantity of product k that agent a_e is scheduled to provide to agent a_j, and v_{ejk} represents the arrival time associated with flow y_{ejk}. Once the disruption occurs, the disrupted agent a_e detects the disruption and realizes the arrival

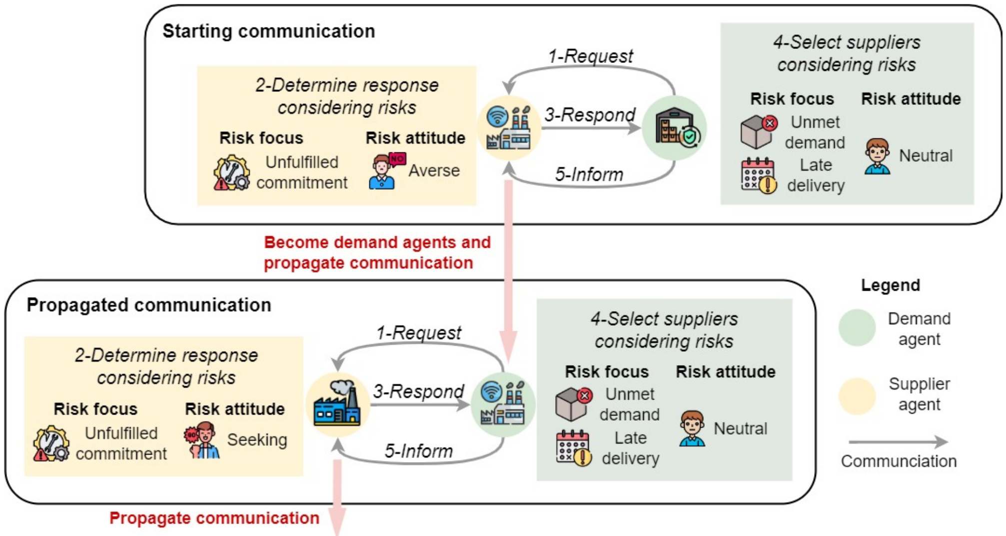

Figure 4.2: The agent communication and decision-making process with risk management for disruption response. The agent risk focus and attitude changes for different agent roles.

times change to $v'_{ejk} > v_{ejk}$. Therefore, the disrupted agent a_e informs all downstream agents a_j about the new arrival time v'_{ejk}.

4.2.2.2 Agent request and response

Request From the original plan, a downstream agent a_j is scheduled to receive product k and use it to produce product k' starting at time $o_{jk'}$. Once agent a_j receives the information about the new arrival time, it checks whether its production is affected by the lead time disruption. If $v'_{ejk} > o_{jk'}$ (i.e., the product k is late), then a_j becomes a demand agent that seeks to obtain product k from alternative supplier agents. Otherwise, a_j can accommodate the disruption and no re-planning decisions are needed. We denote the set of all demand agents as $\mathcal{A}_{dm}$.

Each demand agent $a_j \in \mathcal{A}_{dm}$ sends a request for product k to its upstream suppliers ($a_z \in Z_j(k)$) based on its environment knowledge. The request includes (d_{jk}, t_{jk}), where $d_{jk} = y_{ejk}$ denotes the demand amount for product k, which equals the flow from disrupted agent a_e, and $t_{jk} = o_{jk'}$ is the delivery deadline, which equals the planned production start time.

Response For each agent a_z that receives the request, it needs to make decisions about how it will respond to a_j. The response decisions include $(\bar{y}_z, \bar{v}_z)$, where $\bar{y}_z$ represents the number of products the agent is willing to provide and $\bar{v}_z$ represents the time at which it can deliver the products. In this work, we allow agents to provide production quantities over their nominal production capacity, although these production commitments generally correspond to longer lead times. Specifically, $\bar{y}_z = [\bar{y}^u_{zjk}, \bar{y}^o_{zjk}, \forall a_j \in \mathcal{A}_{dm}, k \in K]^\mathsf{T}$, where $\bar{y}^u_{zjk}$ and $\bar{y}^o_{zjk}$ represents the maximum units of product k that a_z can provide to a_j within and over its production estimations, respectively. The arrival times for $\bar{y}^u_{zjk}$ and $\bar{y}^o_{zjk}$ are different: $\bar{v}_z = [\bar{v}^u_{zjk}, \bar{v}^o_{zjk}, \forall a_j \in \mathcal{A}_{dm}, k \in K]^\mathsf{T}$, where $\bar{v}^u_{zjk}$ and $\bar{v}^o_{zjk}$ represent the arrival times of $\bar{y}^u_{zjk}$ and $\bar{y}^o_{zjk}$ from a_z to a_j, respectively. Note that a_z could receive multiple requests from different demand agents and must make decisions regarding committed quantities accordingly.

We assume a_z has estimations about the amount of product k it can produce, denoted by $\tilde{p}_{zk}$, and the time it can start production, denoted by $\tilde{o}_{zk}$. Both $\tilde{p}_{zk}$ and $\tilde{o}_{zk}$ are random variables with known distributions. We formulate the optimization for the response decision-making as a stochastic programming problem, as shown in (4.4). This model maximizes the weighted sum of income and rewards, subtracting risks, for a response decision:

$$
\begin{aligned}
\max_{\bar{y}_z, \bar{v}_z} \quad & \mathbb{E}[\sum_{a_j \in \mathcal{A}_{dm}, k \in K} r_{zjk}(\bar{y}^u_{zjk} + \bar{y}^o_{zjk}) - w^r \bar{y}^o_{zjk} \\
& \quad + \sum_{a_j \in \mathcal{A}_{dm}} w^p g^p_j \prod_{k \in K} \eta^p_{zjk} + w^t g^t_j \prod_{k \in K} \eta^t_{zjk}] && \text{(4.4a)} \\
\text{s.t.} \quad & \bar{y}^u_{zjk} \leq \mathcal{M} \gamma^u_{zjk}, \forall a_j \in \mathcal{A}_{dm}, k \in K, && \text{(4.4b)} \\
& \bar{y}^o_{zjk} \leq \mathcal{M} \gamma^o_{zjk}, \forall a_j \in \mathcal{A}_{dm}, k \in K, && \text{(4.4c)} \\
& \sum_{a_j \in \mathcal{A}_{dm}} \bar{y}^u_{zjk} + \bar{y}^o_{zjk} \leq \tilde{p}_{zk}, \forall k \in K, && \text{(4.4d)} \\
& \sum_{a_j \in \mathcal{A}_{dm}} \bar{y}^u_{zjk} \leq Q_{zk}, \forall k \in K, && \text{(4.4e)} \\
& (\bar{y}^u_{zjk} + \bar{y}^o_{zjk} - d_{jk}) \eta^p_{zjk} = 0, \forall a_j \in \mathcal{A}_{dm}, k \in K, && \text{(4.4f)} \\
& \bar{y}^u_{zjk} + \bar{y}^o_{zjk} \leq d_{jk}, \forall a_j \in \mathcal{A}_{dm}, k \in K, && \text{(4.4g)} \\
& \bar{v}^u_{zjk} = (\tilde{\ell}_{zjk} + \tilde{o}_{zk}) \gamma^u_{zjk}, \forall a_j \in \mathcal{A}_{dm}, k \in K, && \text{(4.4h)} \\
& t_{jk} \leq \max\{\bar{v}^u_{zjk}, \beta_{zjk} \gamma^o_{zjk} \bar{v}^u_{zjk}\} + \mathcal{M} \eta^t_{zjk}, \forall a_j \in \mathcal{A}_{dm}, k \in K, && \text{(4.4i)} \\
& \eta^t_{zjk}, \eta^p_{zjk}, \gamma^u_{zjk}, \gamma^o_{zjk} \in \{0, 1\}, \forall a_j \in \mathcal{A}_{dm}, k \in K, && \text{(4.4j)}
\end{aligned}
$$

In this model, the objective in the first line of (4.4a) represents the total income that the supplier can earn if its responded flows ($\bar{y}^u_{zjk}+\bar{y}^o_{zjk}$) are selected, and the risk of failing to fulfill the response, which is quantified as the product flows that exceed its production capacity (i.e., $\bar{y}^o_{zjk}$). The second half of the objective function (4.4a) indicates the reward the supplier agent will receive from the demand agents if it can satisfy the demands and deadlines. The parameters g^p_j and g^t_j are the reward gains that agent a_j offers if all the demands and deadlines are satisfied. The binary variables η^p_{zjk} and η^t_{zjk} equal to 1 if the demand and deadline of a_j for product k are satisfied, 0 otherwise. Constraints (4.4b) and (4.4c) indicate the selection of flow response. The binary variable γ^u_{zjk} equals 1 if a_z decides to respond to a_j to provide flow $\bar{y}^u_{zjk}$, and γ^o_{zjk} equal 1 if the response from a_z to a_j includes a production quote that exceeds its production capacity. Constraint (4.4d) indicates that the estimated production is the upper bound of the response, and constraint (4.4e) guarantees that $\bar{y}^u_{zjk}$ does not exceed production capacity. Constraints (4.4f) and (4.4g) indicate whether the response can satisfy the demand. Equation (4.4h) defines the arrival time based on the estimated production start time and lead time. Constraint (4.4i) indicates whether the products can be delivered before the deadline. Constraint (4.4j) represents the range of all binary variables in this model. Note that the arrival times $\bar{v}^o_{zjk}$ of over-capacity product ($\bar{y}^o_{zjk}$) cannot be smaller than the nominal arrival time (i.e., $\bar{v}^o_{zjk} = \beta_{zjk}\bar{v}^u_{zjk} \geq \bar{v}^u_{zjk}$).

To solve this stochastic optimization model, we apply the Sample Average Approximation (SAA) approach [114], which generates a finite realization of the uncertain parameters following a distribution. In this case, the known uncertain parameters include $\tilde{p}_{zk}$, $\tilde{\ell}_{zjk}$, and $\tilde{o}_{zk}$. We denote $\xi_i = [\tilde{p}_{zk,i}, \tilde{\ell}_{zjk,i}, \tilde{o}_{zk,i}, \forall a_j \in \mathcal{A}_{dm}, k \in K]^\intercal$ as a vector of the sampled realizations of all the uncertain parameters. The sampling process follows the distribution of each parameter independently. We sample $\mathcal{Q}$ times, thus the objective value can be calculated as:

$$\mathbb{E}_{1\leq i\leq \mathcal{Q}}[\mathcal{J}_s(\bar{y}_{z,i}, \bar{v}_{z,i})] = \frac{1}{\mathcal{Q}}\sum_{i}^{\mathcal{Q}} \mathcal{J}_s(\bar{y}_{z,i}, \bar{v}_{z,i}) \tag{4.5}$$

and the constraints become the augmentation of all $\mathcal{Q}$ samples. Then we determine the final response $(\bar{y}^*_z, \bar{v}^*_z)$ as the expected value of the responses optimized from all the samples:

$$\bar{y}^*_z = \mathbb{E}_{1\leq i\leq \mathcal{Q}}[\bar{y}_{z,i}],\ \bar{v}^*_z = \mathbb{E}_{1\leq i\leq \mathcal{Q}}[\bar{v}_{z,i}] \tag{4.6}$$

Note that objective (4.5) represents the optimization for a risk-neutral agent to minimize the expected value. A risk-averse agent would be designed to consider the worst-case scenario, where the optimization (4.4) can be formulated as $\min_{1\leq i\leq \mathcal{Q}} \mathcal{J}_s(\bar{y}_z, \bar{v}_z)$.

4.2.2.3 Decision-making for supplier selection

Supplier selection Once the demand agent a_j receives all of the responses from the supplier agents, it makes the decisions for selecting suppliers by solving an optimization problem.

The decisions include the quantity of products each supplier agent can provide, denoted by $\hat{y}_j = [\hat{y}_{zjk}, \forall a_z \in Z_j(k), k \in K]^\mathsf{T}$, where $\hat{y}_{zjk}$ represents the determined number of product k that a_j plans to get from a_z. Though the response information $(\bar{y}_z^*, \bar{v}_z^*)$ is deterministic, we assume a_j has different levels of trust with regards to the responses it received. Trust is quantified as the uncertainty regarding the response that demand agent a_j receives from a given supplier agent. For example, the response from supplier agent a_z includes discrete values for product quantity $\bar{y}_z^*$ and arrival time $\bar{v}_z^*$; however, the demand agent a_j evaluates these variables as random variables because unexpected disturbances and variations in production and travel times exist in the real world. We assume that these distributions follow Gaussian distributions $\mathcal{N}(\bar{y}_z^*, \sigma\bar{y}_z^*)$, where σ represents the trust level and is known based on prior knowledge of the agents. Then a_j evaluates the costs to receive products from each supplier along with their uncertainties about the production and delivery.

The optimization for the supplier selection is given as a stochastic programming problem, which we formulate as follows:

$$\min_{\hat{y}_j} \mathbb{E}[C_d + w_j^t \sum_{k \in K} \Delta_{jk}^t + w_j^p \sum_{k \in K} \Delta_{jk}^p] \tag{4.7a}$$

$$\text{s.t.} \quad C_d = \sum_{a_z \in Z_j(k), k \in K} m_{zjk} \hat{y}_{zjk}, \tag{4.7b}$$

$$\Delta_{jk}^t = \sum_{a_z \in Z_j(k)} \big(\lambda_{zjk}^u \max\{(\bar{v}_{zjk}^u)^* - t_{jk}, 0\} + \lambda_{zjk}^o \max\{(\bar{v}_{zjk}^o)^* - t_{jk}, 0\}\big), \forall k \in K, \tag{4.7c}$$

$$\Delta_{jk}^p = \max\{d_{jk} - \sum_{a_z \in Z_j(k)} \hat{y}_{zjk}, 0\}, \forall k \in K, \tag{4.7d}$$

$$\hat{y}_{zjk} \leq \big((\bar{y}_{zjk}^u)^* + (\bar{y}_{zjk}^o)^*\big) \lambda_{zjk}^u, \forall a_z \in Z_j(k), k \in K, \tag{4.7e}$$

$$\hat{y}_{zjk} - (\bar{y}_{zjk}^u)^* \leq \mathcal{M} \lambda_{zjk}^o, \forall a_z \in Z_j(k), k \in K, \tag{4.7f}$$

$$\lambda_{zjk}^u, \lambda_{zjk}^o \in \{0, 1\}, \forall a_z \in Z_j(k), k \in K, \tag{4.7g}$$

The objective of this program is to minimize the cost (C_d) to purchase the products considering the risk of unmet demand and delivery lateness due to uncertainties, as shown in (4.7a). Equation (4.7b) defines the total cost to obtain flow $\hat{y}_{zjk}$ from the suppliers. Equation (4.7c) defines the total lateness of the product delivery for all the selected suppliers, and equation (4.7d) defines the total unmet demand. Constraint (4.7e) represents that agent a_j cannot select product flows that sum to more than the suppliers' responses. The binary variable λ^u_{zjk} equals 1 if supplier a_j is selected for product k. Constraint (4.7f) indicates whether the selected a_j responds with products that exceed its production capacity.

We use the SAA approach to solve this stochastic optimization model. In this case, the uncertain parameters are $\bar{y}^*_{zjk}$ and $\bar{v}^*_{zjk}$. Similar to our above description, we denote $\eta_i = [\bar{y}^*_{zjk,i}, \bar{v}^*_{zjk,i}, \forall a_z \in Z_j(k), k \in K]^\mathsf{T}$ as a vector of the sampled realizations of the uncertain parameters, with the sampling process following the distributions of each parameter independently. We sample $\mathcal{Q}$ times, thus the objective value can be calculated as:

$$\mathbb{E}_{1 \leq i \leq \mathcal{Q}}[\mathcal{J}_d(\hat{y}_{z,i})] = \frac{1}{\mathcal{Q}} \sum_{i}^{\mathcal{Q}} \mathcal{J}_d(\hat{y}_{z,i}) \tag{4.8}$$

where the constraints become the augmentation of all $\mathcal{Q}$ samples. Then we determine the final supplier selection $\hat{y}^*_z$ as the expected value of the selections optimized from all the samples:

$$\hat{y}^*_z = \mathbb{E}_{1 \leq i \leq \mathcal{Q}}[\hat{y}_{z,i}] \tag{4.9}$$

This presented optimization that minimizes expected value is designed for a risk-neutral agent. When an agent becomes risk-averse, it tends to minimize the cost and risk in the worst-case scenario, thus the optimization (4.7) can be reformulated as $\min_{1 \leq i \leq \mathcal{Q}} \mathcal{J}_d(\hat{y}_z)$.

Inform selection Once the supplier selection decisions are made, all the demand agents a_j inform each selected supplier agent a_z about the new flow plan $\hat{y}_{zjk}$.

4.2.2.4 Communication propagation

Since each selected supplier agent, a_z, commits to providing products to meet the needs of the demand agents, this may introduce additional product/component needs from their suppliers to ensure sufficient products to meet these new commitments. In this case, these selected supplier agents must propagate demand requests in order to meet the needs of their related supplier agents. The propagation process stops when all of the agents have met their additional needs (e.g., the requests have been propagated through all upstream agents in the network). The detailed communication propagation can be found in chapter 3.

4.3 Case Study

4.3.1 Case Study Set-up

The case study in this section uses the same supply chain instance shown in Fig. 3.6 in Chapter 3 with additional time attributes. We add lead times to all of the suppliers and cockpit assemblers, and delivery deadlines to all of the customers (i.e., auto assemblers). To generate the initial optimal product flow plan, we revise the centralized model with the lead time feature [112] and use the optimized solution to initialize the supply chain instance. Note that the initial plan is a deterministic plan, which is shown in Fig. 4.3.

To illustrate the uncertainties in supply chains, we develop a discrete-event simulation framework to evaluate the performance of the plans under lead time uncertainties. This out-of-sample simulation is initialized with the flow plans from the upstream agents that only have outflows, starting at time 0. The stochastic lead time for each outflow is sampled from a known normal probability distribution. Then we obtain the delivery information, including quantity and arrival time, at all the downstream agents that just received the products. Note that a downstream agent may receive multiple types of products as components for its own production. In this case, its own production starts when it receives all the needed components, i.e., the production time depends on

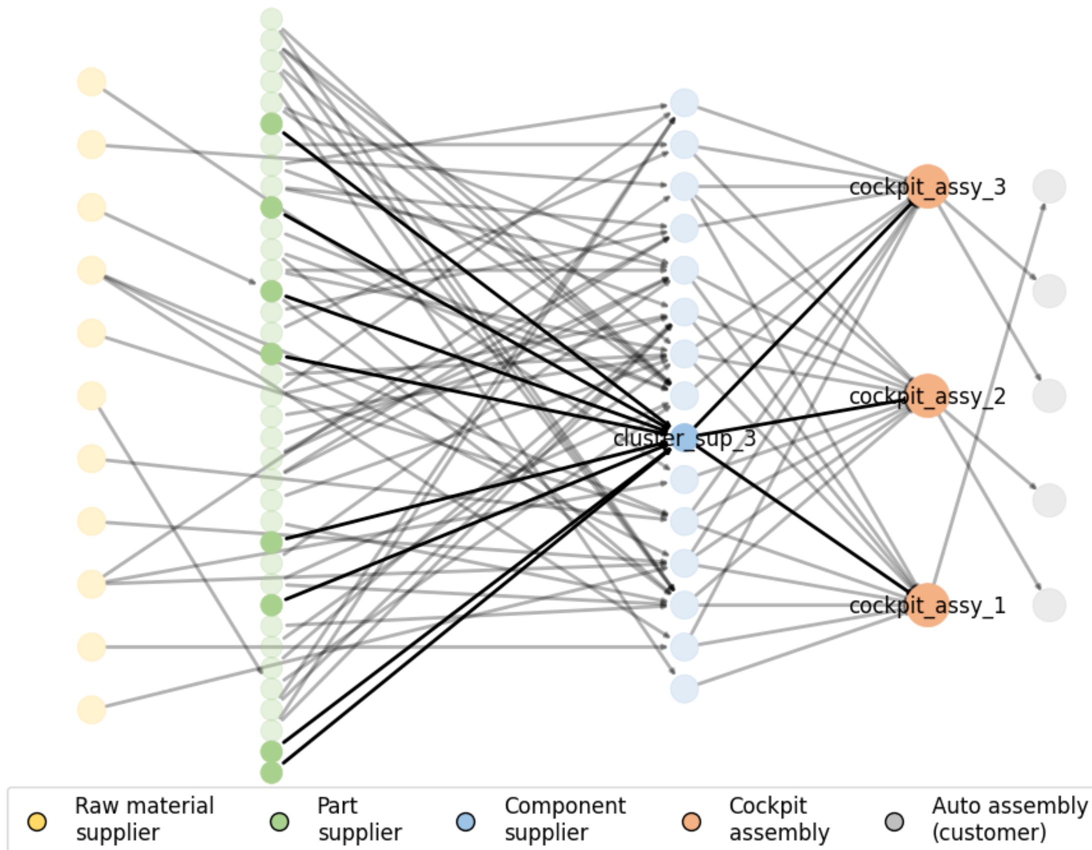

Figure 4.3: The initial plan shown in the supply chain instance. The agents that are affected by the tested disruption are highlighted.

the latest arrival time of needed components. Then the lead time for the downstream agent to deliver products to its downstream agents can be sampled. We continue this process iteratively from upstream to downstream until the final products (i.e., cockpits) are delivered to all customers. The simulation runs multiple times in parallel to analyze the supply chain performance under different realizations of the lead time.

In this work, we consider a disruption that delays product delivery. The disrupted agent is named *cluster_sup_3*, shown as the highlighted blue circle in Fig. 4.3. This agent provides three types of clusters, denoted by a set $K_d = \{cluster\_1, cluster\_2, cluster\_3\}$) to its downstream assembler agents, shown as the highlighted orange circles in Fig. 4.3. For simplicity, we denote these downstream agents as $A_{dm} = \{A1, A2, A3\}$, and *A#* represents *cockpit_sup_#*. Note that there are three other cluster suppliers that could serve as backup agents. We denote all the cluster suppliers as *S1, S2, S3*, and *S4*, and *S#* represents *cluster_sup_#*. Once a disruption is identified, the agent communication strategy is triggered to generate a new plan if necessary, as discussed in Section 4.2.2. Then we run the simulation with the new plan to evaluate the performance of the new plan generated by the proposed approach.

4.3.2 Case Study Results

4.3.2.1 Case study 1: various disruption scales

This case study aims to compare how the proposed approach performs when the disruption impacts the agents at different levels. We consider three disruption scenarios, where the disruption increases the lead time of agent *S3* by 20%, 60%, and 100%. In each disruption scenario, we evaluate the modified plans for instances when the three downstream agents are all risk-neutral and all risk-averse. The out-of-sample simulation runs 300 times in parallel based on the known distributions of the lead time of all the agents. We evaluate the performance by calculating the total delay time for when the downstream agents receive the original production flow. The modified local flows are represented as $[\hat{y}_{zjk}, \forall a_j \in A_{dm}, a_z \in Z_j(k), k \in K_d]^\intercal$, where $\hat{y}_{zjk}$ is the quantity of product k that flows from supplier agent a_z to demand agent a_j. In each simulation round i, we denote the arrival time for flow $\hat{y}_{zjk}$ as $v_{zjk,i}$, thus the lateness of the flow is $\Delta_{zjk,i} = \max\{v_{zjk,i} - t_{jk}, 0\}$, where t_{jk} is the required time for a_j to receive product k. The notation Ω_i represents the set of possible values of $\Delta_{zjk,i}$. We evaluate the performance of the plan by showing the percentage of the products that have the lateness $\Delta_{zjk,i}$ for each simulation round. The product percentage is calculated as the ratio of the total quantities in the flows that are delayed by $\Delta_{zjk,i}$ to the total product quantities:

$$\frac{\sum_{a_j \in A_{dm}, a_z \in Z_j(k), k \in K_d} \hat{y}_{zjk} \text{ if } \Delta_{zjk,i} = \delta_i}{\sum_{a_j \in A_{dm}, a_z \in Z_j(k), k \in K_d} \hat{y}_{zjk}}, \ \forall \delta_i \in \Omega_i \tag{4.10}$$

Therefore, we can get a distribution of these percentages in terms of 300 simulation rounds.

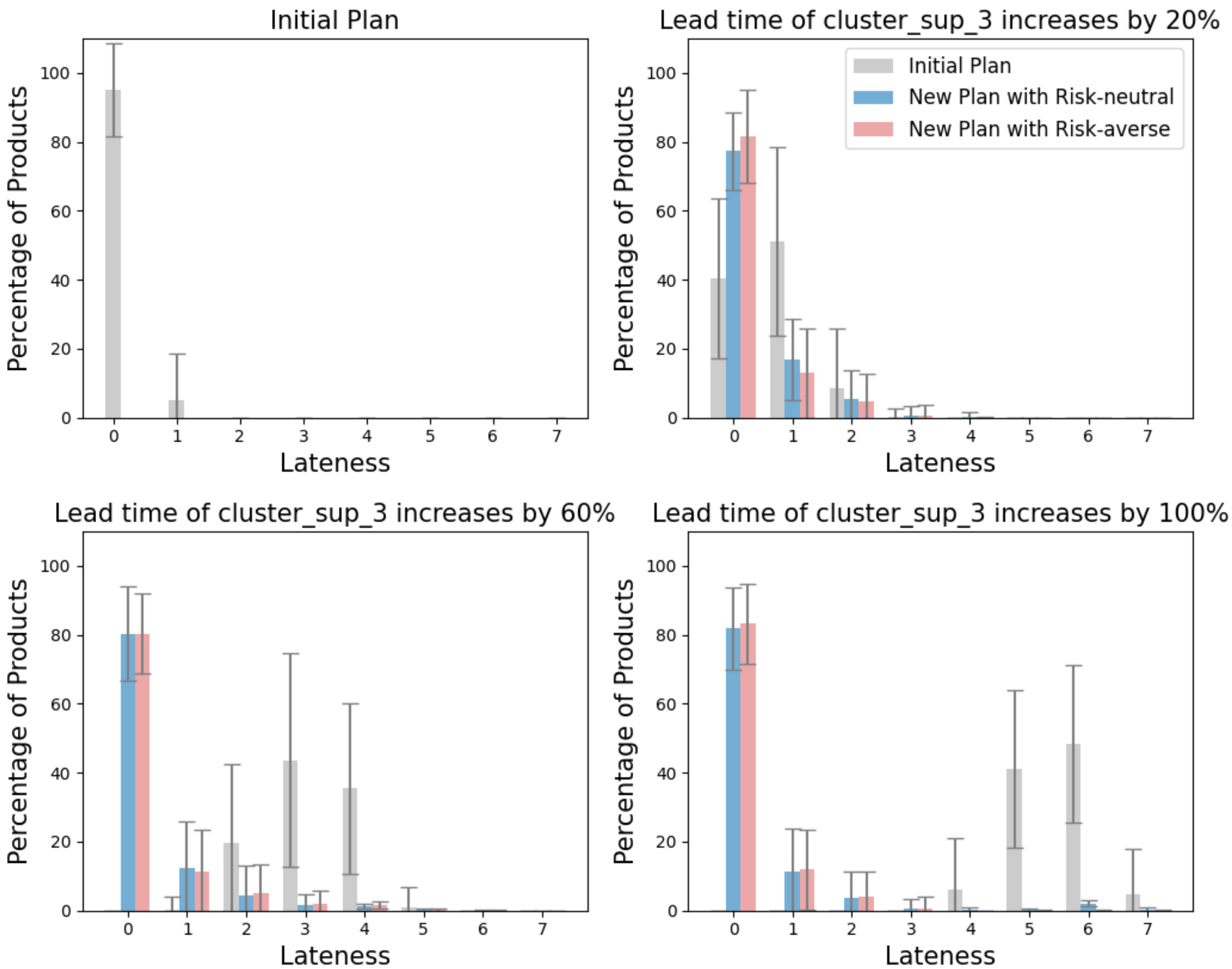

Figure 4.4: The distribution of the percentage of products based on lateness under different disruption and decision-making scenarios

The results shown in Fig. 4.4 indicate that when the disruption increases more lead time, the initial plan leads to a modified plan with more products subject to larger delays. When the proposed re-planning approach is applied, most of the products can be delivered on time or with a small delay. These results demonstrate the potential reduction in delays and overall costs associated with disturbances through the application of a re-planning framework. As expected, the impact of re-planning is more pronounced for larger disruptions.

To investigate the cost of re-planning, we check the objective values of these demand agents that are affected by the disruption. As defined in (4.7a), each demand agent a_j minimizes $\mathcal{J}_d = C_d + w_j^t \sum_{k \in K_d} \Delta_{jk}^t + w_j^p \sum_{k \in K_d} \Delta_{jk}^p$. Note that in this case study, all the demands are satisfied,

Table 4.2: Objective values for different disruption and agent attitudes

Disruption	**0%**	**20%**		**60%**		**100%**	
Risk-attitude	**/**	**Neutral**	**Averse**	**Neutral**	**Averse**	**Neutral**	**Averse**
Product Cost (C_{dm})	31,120	28,386	28,416	28,386	29,342	28,792	29,208
Lateness (L_{dm})	0	1	1	6	5	8	3
Objective Value $\mathcal{J}_{dm} = C_{dm} + w^t L_{dm}$	31,120	128,386	128,416	628,386	529,342	828,792	329,208

i.e., $\sum_{k \in K_d} \Delta^p_{jk} = 0$, thus we focus on the product cost C_d and lateness $\sum_{k \in K_d} \Delta^t_{jk}$. We calculate the total product cost and lateness for the three demand agents:

$$C_{dm} = \sum_{a_j \in \mathcal{A}_{dm}} C_{d,j}, \ L_{dm} = \sum_{a_j \in \mathcal{A}_{dm}} \sum_{k \in K_d} \Delta^t_{jk} \tag{4.11}$$

In addition, the penalty weight for lateness is $w^t = 10^5$ for all the demand agents, thus the total objective can be calculated by $\mathcal{J}_{dm} = C_{dm} + w^t L_{dm}$. Table 4.2 shows the objective values of the initial plan and modified plans for the three disruption scenarios. As mentioned in Section 4.3.1, in the initial plan, all the demand agents obtain clusters from *S3*. This selection is based on the performance of the entire supply chain network, while we focus on the cost and lateness within this subset of agents. Note that Table 4.2 presents the costs and penalties of the deterministic modified plan. Fig. 4.4 shows the results of the simulation where the plan runs with uncertainties.

Compared to the initial plan, the modified plans have lower product costs in all scenarios. This is because *S3* has the lowest lead time and the initial plan tends to minimize the lateness due to the high penalty for lateness, even though *S3* has higher product cost. However, after the disruption occurs, *S3* cannot guarantee on-time delivery. Therefore, Table 4.2 shows that during the re-planning process, these demand agents re-evaluate suppliers and select other suppliers that have lower product costs since lateness is inevitable, at least for a portion of products.

The results also show that as the disruption scale increase, the lateness goes higher, which indicates that the demand agents may still obtain products from the disrupted *S3*. However, when the disruption increases the lead time by 100%, the lateness becomes smaller when the agents are risk-averse. In this case, agents try to minimize the worst case (i.e., potential largest lead time), which mostly occurs if agents choose supplier *S3*. Therefore, agents decide to obtain products from other suppliers, even with the expense of higher production costs. On the other hand, when agents are risk-neutral, they consider the expected value of multiple samples, thus they may still select supplier *S3*, resulting in larger delays. When the disruption scale is smaller, the uncertainties may not lead to a specific worst-case. Consequently, the results from risk-neutral and risk-averse agents could be similar. Note that alternative optimization results may be achieved based on the selection of the applied weighting factors to the cost and delay penalty.

4.3.2.2 Case study 2: heterogeneous risk attitudes

Based on the discussion in case study 1, it is interesting that when the disruption increases the lead time for products from *S3* by 60%, the results for risk-neutral and risk-averse agents are still similar. Since case study 1 presents the total cost and lateness of all the demand agents, it is difficult to specify how the risk attitudes impact supplier selection. Therefore, we conduct tests involving various combinations of risk attitudes for the three demand agents at this disruption scale to check how agents make different supplier selections. Note that the uncertainty when demand agents make decisions comes from a certain level of trust in the responses received from suppliers. As mentioned in Section 4.2.2.2, the demand agents treat the response as normally distributed values, where the response is mean and the trust level affects the variation.

The results are shown in Fig. 4.5, where the widths of the flow arrows are proportional to the quantities of products in the flow, which are labeled near the arrows. Note that in the initial plan, all the flows to the cockpit assemblers are from *S3*. The results show that the demand agents choose to obtain products mainly from other suppliers than *S3*, no matter what risk attitudes of the agents are. This validates the results in Fig. 4.4, where most products are slightly delayed. In general, risk-averse agents obtained fewer products from *S3*.

For product *cluster_1*, agent *A1* decides to switch its main supplier from *S3* to *S1*, regardless of its risk attitude. This decision is driven by several factors. Firstly, the disruption has resulted in an

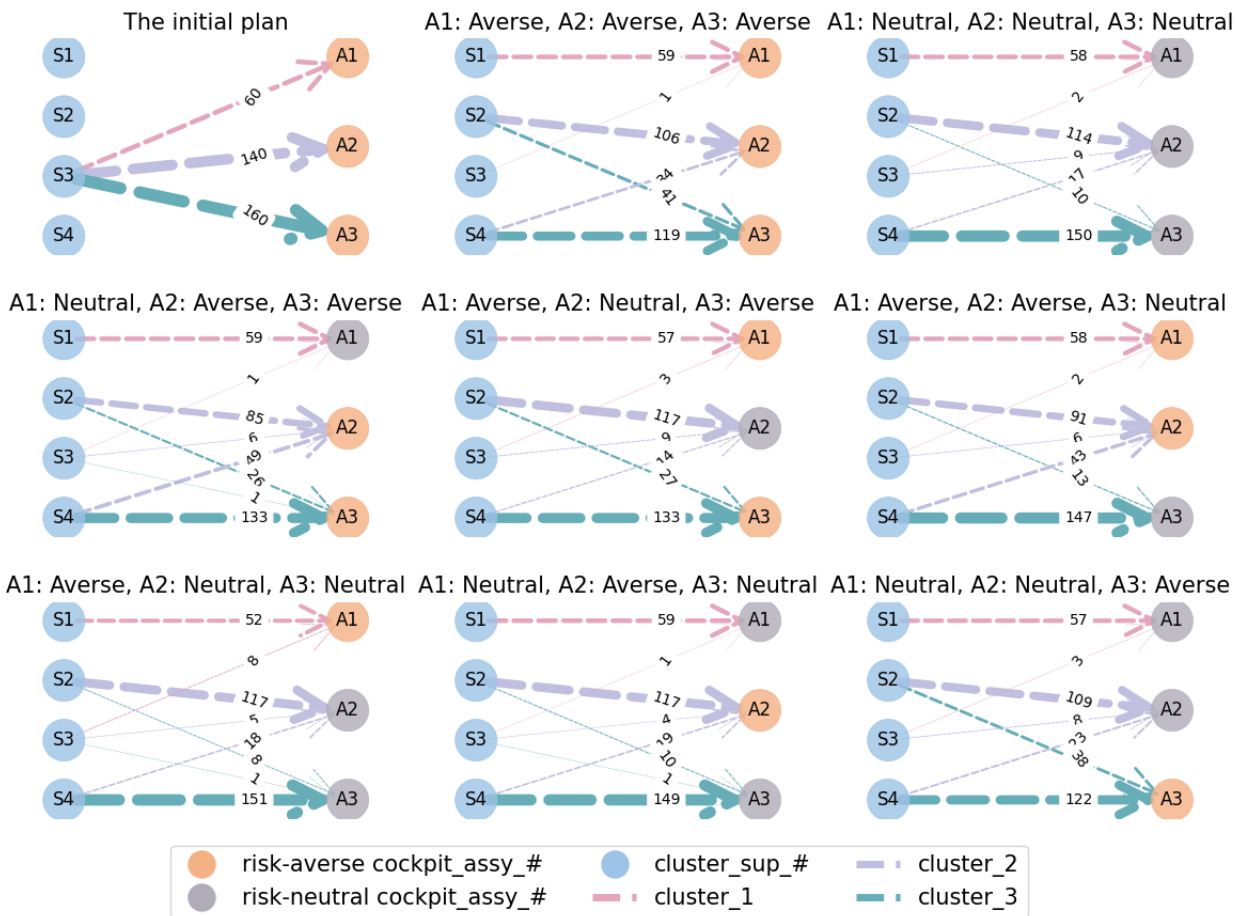

Figure 4.5: The new flow plan for cockpit assemblers to receive clusters when they have different risk attitudes. The numbers on the arrows represent the quantity of products in the flows, and they are proportional to the widths of the arrows. The disruption increases 60% of the lead time for *S3* (i.e., S3 in the figure).

increase in *S3*'s lead time, making it less favorable in terms of timely product delivery. Secondly, *S1* offers a lower cost compared to *S3*. Lastly, *A1* has a higher level of trust in *S1*, meaning that there is a lower level of uncertainty associated with sourcing from *S1*. Considering these factors, *S1* emerges as the preferred choice for agent *A1*, regardless of its risk attitude.

For product *cluster_2*, agent *A2* decides to switch to sourcing from both *S2* and *S4*, with a higher volume of products from *S2*. This is because the nominal lead times of both *S2* and *S4* fulfill the time requirement, but *S2* can produce *cluster_2* with a lower cost. However, the lead time of *S2* is larger than *S4*, which introduces a higher possibility of delay. Therefore, when *A2* is risk-averse, it chooses to increase the number of products obtained from *S4*.

For product *cluster_3*, agent *A3* decides to switch to sourcing from both *S2* and *S4*, with a higher volume of products from *S4*. In this case, *S4* has both a lower cost and a lower lead time. Therefore, *S4* is a preferred supplier, especially when *A3* is risk-neutral. Further, *A3* holds a lower level of trust of *S4*, thus it considers higher uncertainty about *S4*'s response. Therefore, when *A3* is risk-averse and considers the worst case, it chooses to increase the number of products obtained from *S2*.

4.3.3 Discussion and Insights

The case study demonstrates the feasibility and performance of the proposed risk management mechanism. By incorporating uncertainties and risks into the agent optimization, the case study firstly highlights the importance of re-planning in stochastic environments, particularly in the presence of significant disruptions. Furthermore, it elucidates how different risk attitudes influence agent decision-making regarding responses to requests and supplier selection. The proposed framework provides users flexibility to manage heterogeneous risks in their supply chains and evaluate performance-based and self-defined metrics.

This case study can be extended from different perspectives to conduct a more comprehensive investigation of heterogeneous risk management. Firstly, disruptions on different agents can be tested to investigate how the attributes of the disrupted agents affect the decision-making when agents have different risk attitudes. Secondly, while this work focuses on testing different risk attitudes for demand agents, introducing different risk attitudes for supplier agents can contribute to greater heterogeneity in the supply chain network. Additionally, other types of uncertainties, risks, objectives, and weights can be considered to examine how agents strike a balance between risks and objectives. Furthermore, the introduction of additional metrics can enhance the evaluation of the new plans generated by agents, taking into account risk and uncertainty considerations. Expanding the scope of analysis along these lines will enable a deeper understanding and investigation of this heterogeneous risk management mechanism.

4.4 Concluding Remarks

In this chapter, we provide a heterogeneous risk management framework to improve the resiliency of supply chain networks. The heterogeneity includes different agent risk focuses and risk attitudes, depending on the role that agents play and their current status. This framework allows agents to choose different risk-based models to solve their local problems based on their own knowledge, shared information, risk attitudes, and goals to achieve. These models differ in how agents measure the consequences of uncertainties, i.e., how they take into account risk and whether they are risk-averse or risk-neutral. In addition, agents can update their risk focuses and attitudes dynamically as their own attributes and/or the local environment change. Such flexibility in allowing agents to evaluate risk heterogeneously is difficult to be achieved in a centralized approach.

More specifically, we reformulate the agent decision-making in Chapter 3 as stochastic optimization problems by incorporating uncertainties and modeling the risks that agents are interested in. Solving a stochastic optimization instead of the deterministic models, agents can balance their objectives and risks under disruptions at different scales, thus providing a more resilient new plan to respond to disruptions. This could further lead to the possibility of optimizing multi-objective problems under uncertainty using distributed algorithms.

To conclude, we showcase the feasibility of the proposed heterogeneous risk management framework and analyze how the risks affect agent decision-making through two case studies. This heterogeneity acknowledges the reality that different suppliers may have varying risk preferences and risk tolerance levels. By incorporating these factors into the decision-making process, the mechanism provides a more comprehensive and realistic approach to managing risks in supply chains. This can lead to improved decision-making, more rational supplier selection, and ultimately enhanced supply chain resiliency.

CHAPTER 5

Conclusions and Future Directions

Highly dynamic and uncertain industrial environments bring unexpected disruptions with various locations, scales, time periods, and impacts. This disruption variety may require different decision-making strategies for enterprises to handle the disruption efficiently. In this dissertation, we develop a distributed system-level control approach using a multi-agent framework that provides the flexibility to handle various disruptions. Showcased by the application to manufacturing systems and supply chain networks, this approach allows users to determine disruption responses at either the internal factory level or external supply level, depending on disruption attributes and system attributes. In addition, the proposed design of a multi-agent framework is applicable to other complex systems that consist of multiple intelligent entities, such as multi-robot teams. In this chapter, we restate the main contributions, identify future research directions, and discuss further impacts of this dissertation.

5.1 Contributions

The previous chapters highlight the contributions of this dissertation from the application perspectives: 1) a dynamic and resilient rescheduling strategy for manufacturing systems; 2) a heuristic-guided dynamic re-planning strategy for supply chain networks; and 3) a heterogeneous risk management mechanism for resilient disruption response for supply chain networks. In this section, we highlight the dissertation contributions based on the system-level control for industrial applications (e.g., manufacturing systems and supply chains) and describe how flexibility, agility, and resiliency can be improved within these systems.

5.1.1 System-Level Modeling and Control

This dissertation introduces a generalized way to utilize multi-agent systems to model and represent complex industrial environments. Using the proposed design for individual agents, practitioners can capture the heterogeneity of industrial systems. This heterogeneity encompasses a wide range of aspects, including various agent types and capabilities, and distinct decision-making processes. Additionally, the nature of individual agent design provides scalability and reusability to accommodate various types of systems at different scales or adjust existing ones to suit evolving industrial requirements. As a result, it empowers practitioners to conduct simulations and analysis, thereby gaining deeper insights into the system's overall performance and behavior.

Furthermore, this framework provides enterprises with the flexibility to perform system-level control based on specific disruptions and individual agent attributes that best address the nuances of a disruption. Through this agent model, various heuristics can be designed to guide agent communication and decision-making depending on the problems. For example, users may choose to reconfigure the factory plant to recover performance if a machine breaks down. However, if there is a material supply shortage, they may trigger the agent decision-making from the supply chain perspective instead of the factory.

System-level flexibility managed at a local level is difficult to achieve using conventional centralized approaches. The following sections summarize how the proposed design of a multi-agent framework can improve flexibility, agility, and resiliency.

5.1.2 Flexibility: Model-Based Agent Architecture

This dissertation proposes a generalized model-based agent architecture that includes a Knowledge Base, a Communication Manager, and a Decision Manager. The Knowledge Base consists of agent goals, plans, as well as agent states, capabilities, and environments, which are all formulated as various models. These models could be different for different types of agents, such as a discrete event system for RA's capability model and flow dynamics for supplier agent's state model, but this architecture allows agents to dynamically update their intelligence depending on their status. For example, agents can explore the systems to update their knowledge of the environment and can change their decision-making model based on new information obtained from communication. Therefore, this model-based architecture enables agents to react to various scenarios on the fly and improves the system flexibility compared to existing rule-based agents. In addition, this architecture is extensible to different systems and scalable to large and complex systems. The case study of the simulated manufacturing system in Chapter 2 showcases the ability to dynamically reschedule the resources on the fly. The case study of the supply chain instance in Chapter 3 validates the ability to apply the proposed approach to large-scale and various systems, showcasing the flexibility and scalability of the framework toward handling a diverse set of disruptions.

5.1.3 Agility: Heuristic-Based Communication Strategy

This dissertation proposes directed agent communication strategies for solving the rescheduling problem in manufacturing systems and the re-planning problem in supply chain networks. Instead of allowing agents to communicate with all the other agents, the proposed strategies utilize agents' environment knowledge as heuristics to guide agent communication as needed. For manufacturing systems, resource agents (RAs) communicate with each other following a capability-based clustering scheme to avoid unnecessary communications. In supply chain networks, agents retrieve their knowledge about environments to identify the agents they should communicate with, such as upstream agents or downstream agents. Therefore, this heuristic-guided communication strategy enables agents to reduce unnecessary communications and thus improves the system agility compared to existing centralized approaches and fully communicated systems. Showcased by both case studies of a simulated manufacturing system and a supply chain network, the proposed approach enables agents to use local communication to obtain information and make nearly-optimal decisions in most cases. In conclusion, this work makes significant strides in advancing the field of dynamic scheduling and real-time decision-making through the proposed communication strategies.

5.1.4 Resiliency: Optimization-Based Decision-Making Model

This dissertation develops decision-making for various agents using optimization models that incorporate uncertainties and risks. Agents are independent entities that play different roles in the system, thus different optimization models should be applied depending on their status. This dissertation formulates various agent decision-making as optimization models with different quantified risks to cope with uncertainties in the systems. In this way, agents are able to optimize their behaviors heterogeneously based on their own knowledge, objectives, risk attitudes, and roles to improve the system resiliency compared to existing deterministic approaches. The resiliency performance is showcased through simulated case studies. Through the simulation of a manufacturing system and supply chain network in a stochastic setting, it is showcased that the proposed approach enables heterogeneous risk management and provides a resilient disruption response. To conclude, this work pushes the field of distributed stochastic optimization, as well as practical supply chain risk management, when the network becomes complex and large.

5.2 Limitations and Future Work

This dissertation has investigated several promising research areas for the development of distributed intelligence using multi-agent approaches for manufacturing systems and supply chain networks. The feasibility and performance are showcased by several simulated case studies. However, there are still remaining limitations and considerable unexplored questions in this field to make it applicable to the real world. In this section, we briefly discuss four of the potential research directions that can be investigated to realize real-world applications from the perspectives of flexibility, agility, and resiliency.

5.2.1 Flexibility: Multiple Disrupted Agents

Most large-scale disruptions impact multiple agents in supply chain networks. For example, the COVID-19 pandemic disrupted the supply chain with customer demand shifts, factory shutdowns, and transportation limitations. However, this dissertation focuses on the design of rescheduling strategies and the simulated case studies that only consider the disruption of a single agent. Therefore, how agents communicate and make decisions to recover the system performance in the presence of multiple disrupted agents should be addressed through future research efforts.

In scenarios when multiple agents are disrupted, it is important to investigate the relationships between the disrupted agents, as well as the impact of the disruptions on connected agents in either a competitor or collaborator position. By modeling different agent relationships, an extension to the communication strategy in Chapter 3 is to define agent groups depending on whether they can be considered together or have to be handled individually. Once the agent groups are formed, a priority mechanism can be designed to provide heuristics for the communication behaviors. Additionally, to adopt this approach to the industry, both agent collaborative and individual goal-seeking behavior should also be explored considering the complex relationships and privacy issues. This extension would enhance the flexibility of the supply chain network to cope with various disruptions and enable a more practical disruption response strategy.

5.2.2 Agility: Prediction and Learning

Learning and predictive agents have made much headway in the field of computer science but have not been fully utilized for rescheduling and re-planning problems in industrial systems. While the proposed framework enables reactive disruption response, learning techniques can be employed to improve agent initialization and behaviors, such as negotiation strategy and knowledge exploration. For example, the historical data could provide more insightful information to initialize agents, such as their knowledge about themselves (e.g., risk attitudes) and environments (e.g., trust of other agents). This learned knowledge would then equip agents with enhanced heuristics to guide more efficient agent communication and decision-making.

In addition, learning from big data can provide proactive and predictive information to improve the agility of systems. For instance, machine learning could utilize the data to generate adequate disruption scenarios as a database for quick automated disruption identification and estimation. By combining the proposed reactive approaches in Chapter 2 and 3 with the learning-based proactive approach, enterprises are able to build to sophisticated decision-making strategy to provide a more agile response to disruptions.

5.2.3 Resiliency: Hybrid Decision-making

Based on the investigated trade-offs between centralized and distributed approaches in this dissertation, pairing these two approaches to utilize the full potential of both approaches could improve system resiliency. An initial study has been proposed in [44] that allows a centralized controller to provide authority to agents to make decisions and also take over the control if needed. It is important to define under what conditions which approach should be used. For example, the distributed approach could be the optimal strategy if the disruption is on a small scale or when the agents need to explore or identify new disruption strategies. To realize hybrid decision-making, a comprehensive analysis to better quantify the performance of multi-agent control and centralized approaches under different scenarios should be conducted.

This dissertation formulates agent decision-making as an integer programming optimization problem, however, heterogeneous systems may use different techniques to solve different problems. Other problem-solving techniques, such as fuzzy logic and multi-objective optimization, should be investigated.

5.2.4 Implementation: Existing Standard

While the proposed multi-agent framework was evaluated using simulations, there are still a number of challenges that must be addressed in order to implement the framework on a real system. Focusing on high-level scheduling and planning, this dissertation has not investigated low-level control considering the models and dynamics of the physical system. This low-level information may affect high-level decision-making. Furthermore, practical limitations, such as data collection, computing power, and cyber-physical communication latency, should also be considered.

In addition, agent-specific standards such as ISO/International Electrotechnical Commission (IEC) [115] and Foundation for Intelligent Physical Agents (FIPA) [116] should be studied. In high-level agent communication, this dissertation applies Contract Net Protocol (CNP), which aligns with FIPA, to design agent communication. However, the impact of the physical limitations of the information shared through communication has not been considered.

5.3 Outlook and Impact

Overall, this dissertation applies distributed decision-making strategy using a multi-agent framework to enable intelligent industrial systems, which pushes these three research fields: smart industry, multi-agent systems, and distributed optimization.

5.3.1 Smart Industry

From the perspective of industrial applications, this dissertation provides a new unified framework to manage enterprises' internal factories and external supply chains in a distributed way, enabling system-level intelligence and autonomy. Users can apply the proposed work to model and control their manufacturing systems and supply chain network with enhanced flexibility, agility, and resiliency in dynamic industrial environments. Specifically for disruption responses, this work can compute a good recovery plan quickly to respond to the environments. Therefore, enterprises may become more willing to accept customized production orders, participate in more connected supply chain networks, and integrate new equipment and technologies. In addition, the proposed framework can be integrated into a decision support system to conduct various simulations, which helps enterprises analyze the system performance under different conditions that could be too complex and risky to be tested in the real world.

5.3.2 Multi-agent Systems

The multi-agent community has identified a common issue of lacking benchmarks and standardization. This dissertation provides a way to standardize the multi-agent system design for distributed intelligence and successfully applies this approach to two different systems: manufacturing systems and supply chain networks. Since the baseline of the framework is not system-specific, this framework can be extended to other complex systems, such as multi-robot teams and autonomous vehicles, that consist of multiple intelligent and autonomous entities. For example, for a system of multiple Autonomous Guided Vehicles (AGVs) in a warehouse, each AGV can be designed following the proposed framework. Specifically, an AGV agent's knowledge base could include robot dynamics as a state model, planned movement as intentions, warehouse maps as an environment model, etc. Based on the knowledge, agents can communicate to solve (re)planning problems, such as task re-allocation when there are AGVs unavailable. In addition, this standardized agent framework also lays a foundation for identifying typical metrics to evaluate the performance of distributed agent-based systems.

5.3.3 Distributed Optimization

A significant amount of traditional scheduling and planning problems are NP-hard, especially when considering uncertainties and multiple objectives. Though this dissertation focuses on rescheduling problems, the proposed agent communication with stochastic optimization leads to the possibility of heuristic-guided distributed algorithms to provide a solution to these NP-hard problems. In addition, applying this multi-agent framework may enable a more flexible decomposition of the global problem. Instead of decomposing the global objective function to the sum of several sub-functions, assigning local optimization allows agents to solve agent conflicts, as well as model the collaborative and competitive relationship between agents.

BIBLIOGRAPHY

[1] (2022) Share of economic sectors in the global gross domestic product (GDP) from 2011 to 2021. World Bank. [Online]. Available: https://www.statista.com/statistics/256563/share-of-economic-sectors-in-the-global-gross-domestic-product/

[2] (2021) Employment in industry (% of total employment) (modeled ILO estimate). World Bank. [Online]. Available: https://data.worldbank.org/indicator/SL.IND.EMPL.ZS?contextual=default

[3] P. Leitão, "Agent-based distributed manufacturing control: A state-of-the-art survey," *Engineering applications of artificial intelligence*, vol. 22, no. 7, pp. 979–991, 2009.

[4] S. J. Hu, "Evolving paradigms of manufacturing: From mass production to mass customization and personalization," *Procedia Cirp*, vol. 7, pp. 3–8, 2013.

[5] M. Ben-Daya, E. Hassini, and Z. Bahroun, "Internet of things and supply chain management: a literature review," *International Journal of Production Research*, vol. 57, no. 15-16, pp. 4719–4742, 2019.

[6] I. Kovalenko, "Intelligent product agents for multi-agent control of manufacturing systems," Ph.D. dissertation, University of Michigan, 2020.

[7] G. Baryannis, S. Validi, S. Dani, and G. Antoniou, "Supply chain risk management and artificial intelligence: state of the art and future research directions," *International Journal of Production Research*, vol. 57, no. 7, pp. 2179–2202, 2019.

[8] J.-H. Lee and C.-O. Kim, "Multi-agent systems applications in manufacturing systems and supply chain management: a review paper," *International Journal of Production Research*, vol. 46, no. 1, pp. 233–265, 2008.

[9] T. Rahman, F. Taghikhah, S. K. Paul, N. Shukla, and R. Agarwal, "An agent-based model for supply chain recovery in the wake of the covid-19 pandemic," *Computers & Industrial Engineering*, vol. 158, p. 107401, 2021.

[10] A. D. Ganesh and P. Kalpana, "Future of artificial intelligence and its influence on supply chain risk management–a systematic review," *Computers & Industrial Engineering*, p. 108206, 2022.

[11] M. Bi, I. Kovalenko, D. M. Tilbury, and K. Barton, "Dynamic resource allocation using multi-agent control for manufacturing systems," *IFAC-PapersOnLine*, vol. 54, no. 20, pp. 488–494, 2021.

[12] (2021) The pandemic's home-workout revolution may be here to stay. Washington Post. [Online]. Available: https://www.washingtonpost.com/road-to-recovery/2021/01/07/home-fitness-boom/

[13] (2020) How gm and ford switched out pickup trucks for breathing machines. The Verge. [Online]. Available: https://www.theverge.com/2020/4/15/21222219/general-motors-ventec-ventilators-ford-tesla-coronavirus-covid-19

[14] (2020) Dumped milk, smashed eggs, plowed vegetables: Food waste of the pandemic. New York Times. [Online]. Available: https://www.nytimes.com/2020/04/11/business/coronavirus-destroying-food.html?smid=tw-share

[15] C. W. Craighead, J. Blackhurst, M. J. Rungtusanatham, and R. B. Handfield, "The severity of supply chain disruptions: design characteristics and mitigation capabilities," *Decision sciences*, vol. 38, no. 1, pp. 131–156, 2007.

[16] F. Lücker, R. W. Seifert, and I. Biçer, "Roles of inventory and reserve capacity in mitigating supply chain disruption risk," *International Journal of Production Research*, vol. 57, no. 4, pp. 1238–1249, 2019.

[17] M. Bi, G. Chen, D. M. Tilbury, S. Shen, and K. Barton, "A model-based multi-agent framework to enable an agile response to supply chain disruptions," in *2022 IEEE 18th International Conference on Automation Science and Engineering (CASE)*. IEEE, 2022, pp. 235–241.

[18] F. Lopez, Y. Shao, Z. M. Mao, J. Moyne, K. Barton, and D. Tilbury, "A software-defined framework for the integrated management of smart manufacturing systems," *Manufacturing Letters*, vol. 15, pp. 18–21, 2018.

[19] A. R. Boccella, P. Centobelli, R. Cerchione, T. Murino, and R. Riedel, "Evaluating centralized and heterarchical control of smart manufacturing systems in the era of industry 4.0," *Applied Sciences*, vol. 10, no. 3, p. 755, 2020.

[20] S. Croom, P. Romano, and M. Giannakis, "Supply chain management: an analytical framework for critical literature review," *European journal of purchasing & supply management*, vol. 6, no. 1, pp. 67–83, 2000.

[21] R. J. Abumaizar and J. A. Svestka, "Rescheduling job shops under random disruptions," *International journal of production research*, vol. 35, no. 7, pp. 2065–2082, 1997.

[22] K. Ransikarbum, R. Pitakaso, and N. Kim, "A decision-support model for additive manufacturing scheduling using an integrative analytic hierarchy process and multi-objective optimization," *Applied Sciences*, vol. 10, no. 15, p. 5159, 2020.

[23] X. Liu, F. Chu, F. Zheng, C. Chu, and M. Liu, "Parallel machine scheduling with stochastic release times and processing times," *International Journal of Production Research*, vol. 59, no. 20, pp. 6327–6346, 2021.

[24] J.-H. Lee and H.-J. Kim, "Reinforcement learning for robotic flow shop scheduling with processing time variations," *International Journal of Production Research*, vol. 60, no. 7, pp. 2346–2368, 2022.

[25] R. Liu, R. Piplani, and C. Toro, "Deep reinforcement learning for dynamic scheduling of a flexible job shop," *International Journal of Production Research*, vol. 60, no. 13, pp. 4049–4069, 2022.

[26] C. Morariu, O. Morariu, S. Răileanu, and T. Borangiu, "Machine learning for predictive scheduling and resource allocation in large scale manufacturing systems," *Computers in Industry*, vol. 120, p. 103244, 2020.

[27] S. Yang and Z. Xu, "Intelligent scheduling and reconfiguration via deep reinforcement learning in smart manufacturing," *International Journal of Production Research*, vol. 60, no. 16, pp. 4936–4953, 2022.

[28] J. Fischer, C. Lieberoth-Leden, J. Fottner, and B. Vogel-Heuser, "Design, application, and evaluation of a multiagent system in the logistics domain," *IEEE Transactions on Automation Science and Engineering*, vol. 17, no. 3, pp. 1283–1296, 2020.

[29] M. Giannakis and M. Louis, "A multi-agent based system with big data processing for enhanced supply chain agility," *Journal of Enterprise Information Management*, 2016.

[30] X. Chen, Y.-S. Ong, P.-S. Tan, N. Zhang, and Z. Li, "Agent-based modeling and simulation for supply chain risk management-a survey of the state-of-the-art," in *2013 IEEE International Conference on Systems, Man, and Cybernetics*. IEEE, 2013, pp. 1294–1299.

[31] J. Barbosa, P. Leitão, E. Adam, and D. Trentesaux, "Dynamic self-organization in holonic multi-agent manufacturing systems: The adacor evolution," *Computers in industry*, vol. 66, pp. 99–111, 2015.

[32] L. Xu, S. Mak, and A. Brintrup, "Will bots take over the supply chain? revisiting agent-based supply chain automation," *International Journal of Production Economics*, vol. 241, p. 108279, 2021.

[33] S. Xu, X. Zhang, L. Feng, and W. Yang, "Disruption risks in supply chain management: a literature review based on bibliometric analysis," *International Journal of Production Research*, vol. 58, no. 11, pp. 3508–3526, 2020.

[34] P. Leitao, S. Karnouskos, L. Ribeiro, J. Lee, T. Strasser, and A. W. Colombo, "Smart agents in industrial cyber–physical systems," *Proceedings of the IEEE*, vol. 104, no. 5, pp. 1086–1101, 2016.

[35] A. Kumar, S. Luthra, S. K. Mangla, and Y. Kazançoğlu, "Covid-19 impact on sustainable production and operations management," *Sustainable Operations and Computers*, vol. 1, pp. 1–7, 2020.

[36] X. Li, B. Wang, C. Liu, T. Freiheit, and B. I. Epureanu, "Intelligent manufacturing systems in covid-19 pandemic and beyond: framework and impact assessment," *Chinese Journal of Mechanical Engineering*, vol. 33, no. 1, pp. 1–5, 2020.

[37] M. Giannakis and M. Louis, "A multi-agent based framework for supply chain risk management," *Journal of Purchasing and Supply Management*, vol. 17, no. 1, pp. 23–31, 2011.

[38] S. Zhang and T. N. Wong, "Flexible job-shop scheduling/rescheduling in dynamic environment: a hybrid mas/aco approach," *International Journal of Production Research*, vol. 55, no. 11, pp. 3173–3196, 2017.

[39] W. Ho, T. Zheng, H. Yildiz, and S. Talluri, "Supply chain risk management: a literature review," *International Journal of Production Research*, vol. 53, no. 16, pp. 5031–5069, 2015.

[40] D. Ivanov, A. Dolgui, B. Sokolov, and M. Ivanova, "Literature review on disruption recovery in the supply chain," *International Journal of Production Research*, vol. 55, no. 20, pp. 6158–6174, 2017.

[41] E. Sisinni, A. Saifullah, S. Han, U. Jennehag, and M. Gidlund, "Industrial internet of things: Challenges, opportunities, and directions," *IEEE transactions on industrial informatics*, vol. 14, no. 11, pp. 4724–4734, 2018.

[42] J. Moyne, Y. Qamsane, E. C. Balta, I. Kovalenko, J. Faris, K. Barton, and D. M. Tilbury, "A requirements driven digital twin framework: Specification and opportunities," *IEEE Access*, vol. 8, pp. 107 781–107 801, 2020.

[43] D. Ivanov, A. Dolgui, and B. Sokolov, "The impact of digital technology and industry 4.0 on the ripple effect and supply chain risk analytics," *International Journal of Production Research*, vol. 57, no. 3, pp. 829–846, 2019.

[44] I. Kovalenko, J. Moyne, M. Bi, E. C. Balta, W. Ma, Y. Qamsane, X. Zhu, Z. M. Mao, D. M. Tilbury, and K. Barton, "Toward an automated learning control architecture for cyber-physical manufacturing systems," *IEEE Access*, vol. 10, pp. 38 755–38 773, 2022.

[45] W. Shen, L. Wang, and Q. Hao, "Agent-based distributed manufacturing process planning and scheduling: a state-of-the-art survey," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 36, no. 4, pp. 563–577, 2006.

[46] R. Toorajipour, V. Sohrabpour, A. Nazarpour, P. Oghazi, and M. Fischl, "Artificial intelligence in supply chain management: A systematic literature review," *Journal of Business Research*, vol. 122, pp. 502–517, 2021.

[47] G. Cavone, M. Dotoli, N. Epicoco, D. Morelli, and C. Seatzu, "Design of modern supply chain networks using fuzzy bargaining game and data envelopment analysis," *IEEE Transactions on Automation Science and Engineering*, vol. 17, no. 3, pp. 1221–1236, 2020.

[48] W. Shen and D. H. Norrie, "Agent-based systems for intelligent manufacturing: a state-of-the-art survey," *Knowledge and information systems*, vol. 1, pp. 129–156, 1999.

[49] M. Wooldridge, *An introduction to multiagent systems*. John wiley & sons, 2009.

[50] I. Kovalenko, D. Tilbury, and K. Barton, "The model-based product agent: A control oriented architecture for intelligent products in multi-agent manufacturing systems," *Control Engineering Practice*, vol. 86, pp. 105–117, 2019.

[51] I. Kovalenko, D. Ryashentseva, B. Vogel-Heuser, D. Tilbury, and K. Barton, "Dynamic resource task negotiation to enable product agent exploration in multi-agent manufacturing systems," *IEEE Robotics and Automation Letters*, vol. 4, no. 3, pp. 2854–2861, 2019.

[52] S. Adhau, M. L. Mittal, and A. Mittal, "A multi-agent system for distributed multi-project scheduling: An auction-based negotiation approach," *Engineering Applications of Artificial Intelligence*, vol. 25, no. 8, pp. 1738–1751, 2012.

[53] A. Maoudj, B. Bouzouia, A. Hentout, A. Kouider, and R. Toumi, "Distributed multi-agent scheduling and control system for robotic flexible assembly cells," *Journal of intelligent manufacturing*, vol. 30, pp. 1629–1644, 2019.

[54] A. Fiedler, "An agent-based negotiation protocol for supply chain finance," *Computers & Industrial Engineering*, vol. 168, p. 108136, 2022.

[55] M. Hajian Heidary, "Agent-based simulation-optimization model for a bi-objective stochastic multi-period supply chain design problem," *Journal of Industrial Engineering and Management Studies*, vol. 8, no. 2, pp. 175–195, 2022.

[56] P. Leitão and F. Restivo, "Adacor: A holonic architecture for agile and adaptive manufacturing control," *Computers in industry*, vol. 57, no. 2, pp. 121–130, 2006.

[57] P. Ghadimi, C. Wang, M. K. Lim, and C. Heavey, "Intelligent sustainable supplier selection using multi-agent technology: Theory and application for industry 4.0 supply chains," *Computers & Industrial Engineering*, vol. 127, pp. 588–600, 2019.

[58] N. Shukla and S. Kiridena, "A fuzzy rough sets-based multi-agent analytics framework for dynamic supply chain configuration," *International Journal of Production Research*, vol. 54, no. 23, pp. 6984–6996, 2016.

[59] W. Lepuschitz, A. Zoitl, M. Vallée, and M. Merdan, "Toward self-reconfiguration of manufacturing systems using automation agents," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 41, no. 1, pp. 52–69, 2010.

[60] T. Wong, C. Leung, K. Mak, and R. Fung, "Integrated process planning and scheduling/rescheduling—an agent-based approach," *International Journal of Production Research*, vol. 44, no. 18-19, pp. 3627–3655, 2006.

[61] N. Antzoulatos, E. Castro, L. de Silva, A. D. Rocha, S. Ratchev, and J. Barata, "A multi-agent framework for capability-based reconfiguration of industrial assembly systems," *International Journal of Production Research*, vol. 55, no. 10, pp. 2950–2960, 2017.

[62] N. Rodrigues, E. Oliveira, and P. Leitão, "Decentralized and on-the-fly agent-based service reconfiguration in manufacturing systems," *Computers in Industry*, vol. 101, pp. 81–90, 2018.

[63] H.-S. Park and N.-H. Tran, "An autonomous manufacturing system based on swarm of cognitive agents," *Journal of Manufacturing Systems*, vol. 31, no. 3, pp. 337–348, 2012.

[64] F. Maturana, W. Shen, and D. H. Norrie, "Metamorph: an adaptive agent-based architecture for intelligent manufacturing," *International Journal of Production Research*, vol. 37, no. 10, pp. 2159–2173, 1999.

[65] J. Barata and L. M. Camarinha-Matos, "Coalitions of manufacturing components for shop floor agility-the cobasa architecture," *International journal of networking and virtual organisations*, vol. 2, no. 1, pp. 50–77, 2003.

[66] M. Bi, I. Kovalenko, D. M. Tilbury, and K. Barton, "Dynamic distributed decision-making for resilient resource reallocation in disrupted manufacturing systems," *International Journal of Production Research*, pp. 1–21, 2023.

[67] S. Torabi, M. Baghersad, and S. Mansouri, "Resilient supplier selection and order allocation under operational and disruption risks," *Transportation Research Part E: Logistics and Transportation Review*, vol. 79, pp. 22–48, 2015.

[68] M. F. Blos, R. M. da Silva, and H.-M. Wee, "A framework for designing supply chain disruptions management considering productive systems and carrier viewpoints," *International Journal of Production Research*, vol. 56, no. 15, pp. 5045–5061, 2018.

[69] K. Pal and B. Karakostas, "A multi agent-based service framework for supply chain management," *Procedia computer science*, vol. 32, pp. 53–60, 2014.

[70] G. Wang, T. Wong, and X. Wang, "An ontology based approach to organize multi-agent assisted supply chain negotiations," *Computers & Industrial Engineering*, vol. 65, no. 1, pp. 2–15, 2013.

[71] C. S. Tang, "Robust strategies for mitigating supply chain disruptions," *International Journal of Logistics: Research and Applications*, vol. 9, no. 1, pp. 33–45, 2006.

[72] B. Tomlin, "On the value of mitigation and contingency strategies for managing supply chain disruption risks," *Management science*, vol. 52, no. 5, pp. 639–657, 2006.

[73] O. Kwon, G. P. Im, and K. C. Lee, "Mace-scm: A multi-agent and case-based reasoning collaboration mechanism for supply chain management under supply and demand uncertainties," *Expert systems with applications*, vol. 33, no. 3, pp. 690–705, 2007.

[74] P. P. Datta and M. G. Christopher, "Information sharing and coordination mechanisms for managing uncertainty in supply chains: a simulation study," *International Journal of Production Research*, vol. 49, no. 3, pp. 765–803, 2011.

[75] J. Zhang, S. P. Sethi, T.-M. Choi, and T. Cheng, "Pareto optimality and contract dependence in supply chain coordination with risk-averse agents," *Production and Operations Management*, vol. 31, no. 6, pp. 2557–2570, 2022.

[76] Y.-W. Zhou, J. Li, and Y. Zhong, "Cooperative advertising and ordering policies in a two-echelon supply chain with risk-averse agents," *Omega*, vol. 75, pp. 97–117, 2018.

[77] L. Zang, M. Liu, Z. Wang, and D. Wen, "Coordinating a two-stage supply chain with external failure cost-sharing and risk-averse agents," *Journal of Cleaner Production*, vol. 334, p. 130012, 2022.

[78] T.-M. Choi, C. Ma, B. Shen, and Q. Sun, "Optimal pricing in mass customization supply chains with risk-averse agents and retail competition," *Omega*, vol. 88, pp. 150–161, 2019.

[79] B. Fu, W. Smith, D. Rizzo, M. Castanier, and K. Barton, "Heterogeneous vehicle routing and teaming with gaussian distributed energy uncertainty," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 4315–4322.

[80] B. Fu, W. Smith, D. M. Rizzo, M. Castanier, M. Ghaffari, and K. Barton, "Robust task scheduling for heterogeneous robot teams under capability uncertainty," *IEEE Transactions on Robotics*, vol. 39, no. 2, pp. 1087–1105, 2023.

[81] I.-B. Park, J. Huh, J. Kim, and J. Park, "A reinforcement learning approach to robust scheduling of semiconductor manufacturing facilities," *IEEE Transactions on Automation Science and Engineering*, vol. 17, no. 3, pp. 1420–1431, 2019.

[82] J. A. Palombarini and E. C. Martínez, "End-to-end on-line rescheduling from gantt chart images using deep reinforcement learning," *International Journal of Production Research*, pp. 1–30, 2021.

[83] I. R. Uhlmann, R. M. Zanella, and E. M. Frazzon, "Hybrid flow shop rescheduling for contract manufacturing services," *International Journal of Production Research*, vol. 60, no. 3, pp. 1069–1085, 2022.

[84] G. E. Vieira, J. W. Herrmann, and E. Lin, "Rescheduling manufacturing systems: a framework of strategies, policies, and methods," *Journal of scheduling*, vol. 6, no. 1, pp. 39–62, 2003.

[85] A. M. Farid and L. Ribeiro, "An axiomatic design of a multiagent reconfigurable mechatronic system architecture," *IEEE Transactions on Industrial Informatics*, vol. 11, no. 5, pp. 1142–1155, 2015.

[86] S. Rehberger, L. Spreiter, and B. Vogel-Heuser, "An agent approach to flexible automated production systems based on discrete and continuous reasoning," in *2016 IEEE International Conference on Automation Science and Engineering (CASE)*. IEEE, 2016, pp. 1249–1256.

[87] Y. G. Kim, S. Lee, J. Son, H. Bae, and B. Do Chung, “Multi-agent system and reinforcement learning approach for distributed intelligence in a flexible smart manufacturing system,” *Journal of Manufacturing Systems*, vol. 57, pp. 440–450, 2020.

[88] G. Mejía and D. Lefebvre, “Robust scheduling of flexible manufacturing systems with unreliable operations and resources,” *International Journal of Production Research*, vol. 58, no. 21, pp. 6474–6492, 2020.

[89] C. Lee, Y. Lv, and Z. Hong, “Risk modelling and assessment for distributed manufacturing system,” *International Journal of Production Research*, vol. 51, no. 9, pp. 2652–2666, 2013.

[90] M. T. Anbarani, E. C. Balta, R. Meira-Góes, and I. Kovalenko, “Risk-averse model predictive control for priced timed automata,” *arXiv preprint arXiv:2210.15604*, 2022.

[91] J. Klöber-Koch, S. Braunreuther, and G. Reinhart, “Predictive production planning considering the operative risk in a manufacturing system,” *Procedia CIRP*, vol. 63, pp. 360–365, 2017.

[92] G. Guo and S. M. Ryan, “Sequencing mixed-model assembly lines with risk-averse stochastic mixed-integer programming,” *International Journal of Production Research*, vol. 60, no. 12, pp. 3774–3791, 2022.

[93] J. M. Framinan, V. Fernandez-Viagas, and P. Perez-Gonzalez, “Using real-time information to reschedule jobs in a flowshop with variable processing times,” *Computers & Industrial Engineering*, vol. 129, pp. 113–125, 2019.

[94] N. Howden, R. Rönnquist, A. Hodgson, and A. Lucas, “Jack intelligent agents-summary of an agent infrastructure,” in *5th International conference on autonomous agents*, vol. 142, 2001.

[95] I. Kovalenko, E. C. Balta, D. M. Tilbury, and K. Barton, “Cooperative product agents to improve manufacturing system flexibility: A model-based decision framework,” *IEEE Transactions on Automation Science and Engineering*, vol. 20, no. 1, pp. 440–457, 2022.

[96] Y. Qamsane, E. C. Balta, J. Moyne, D. Tilbury, and K. Barton, “Dynamic rerouting of cyber-physical production systems in response to disruptions based on sdc framework,” in *2019 American Control Conference (ACC)*. IEEE, 2019, pp. 3650–3657.

[97] M. Toothman, B. Braun, S. J. Bury, M. Dessauer, K. Henderson, R. Wright, D. M. Tilbury, J. Moyne, and K. Barton, “Trend-based repair quality assessment for industrial rotating equipment,” in *2021 American Control Conference (ACC)*. IEEE, 2021, pp. 502–507.

[98] E. C. Balta, I. Kovalenko, I. A. Spiegel, D. M. Tilbury, and K. Barton, “Model predictive control of priced timed automata encoded with first-order logic,” *IEEE Transactions on Control Systems Technology*, vol. 30, no. 1, pp. 352–359, 2021.

[99] C. M. Macal and M. J. North, “Introduction to agent-based modeling and simulation,” in *Proceedings of the MCS LANS Informal Seminar*, 2006.

[100] H. J. Yoon and W. Shen, "A multiagent-based decision-making system for semiconductor wafer fabrication with hard temporal constraints," *IEEE Transactions on Semiconductor Manufacturing*, vol. 21, no. 1, pp. 83–91, 2008.

[101] M. Bi, D. M. Tilbury, S. Shen, and K. Barton, "A distributed approach for agile supply chain decision-making based on network attributes," *IEEE Transactions on Automation Science and Engineering (Submitted)*, 2023. [Online]. Available: https://drive.google.com/file/d/1kiGmSjlXGwBLUiDibINqjOXXWF2ABikL/view?usp=sharing

[102] M. Parhi, S. Perera, M. Piraveenan, and D. Kasthurirathna, "Topological structure of manufacturing industry supply chain networks," *Complexity*, 2018.

[103] Y. Kim, Y.-S. Chen, and K. Linderman, "Supply network disruption and resilience: A network structural perspective," *Journal of operations Management*, vol. 33, pp. 43–59, 2015.

[104] Y. Kim, T. Y. Choi, T. Yan, and K. Dooley, "Structural investigation of supply networks: A social network analysis approach," *Journal of operations management*, vol. 29, no. 3, pp. 194–211, 2011.

[105] A. Nair and J. M. Vidal, "Supply network topology and robustness against disruptions–an investigation using multi-agent model," *International Journal of Production Research*, vol. 49, no. 5, pp. 1391–1404, 2011.

[106] P. Zhao, Z. Li, X. Han, and X. Duan, "Supply chain network resilience by considering disruption propagation: Topological and operational perspectives," *IEEE Systems Journal*, vol. 16, no. 4, pp. 5305–5316, 2022.

[107] A. Whitbrook, Q. Meng, and P. W. Chung, "Reliable, distributed scheduling and rescheduling for time-critical, multiagent systems," *IEEE Transactions on Automation Science and Engineering*, vol. 15, no. 2, pp. 732–747, 2017.

[108] K. Khalili-Damghani and P. Ghasemi, "Uncertain centralized/decentralized production-distribution planning problem in multi-product supply chains: fuzzy mathematical optimization approaches," *Industrial engineering and management systems*, vol. 15, no. 2, pp. 156–172, 2016.

[109] M. Alemany, A. Esteso, Á. Ortiz, and M. del Pino, "Centralized and distributed optimization models for the multi-farmer crop planning problem under uncertainty: Application to a fresh tomato argentinean supply chain case study," *Computers & Industrial Engineering*, vol. 153, p. 107048, 2021.

[110] R. G. Smith, "The contract net protocol: High-level communication and control in a distributed problem solver," *IEEE Transactions on computers*, vol. 29, no. 12, pp. 1104–1113, 1980.

[111] S. Poslad, "Specifying protocols for multi-agent systems interaction," *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, vol. 2, no. 4, pp. 15–es, 2007.

[112] J.-A. Estrada-Garcia, M. Bi, D. M. Tilbury, K. Barton, and S. Shen, “A multi-objective mixed-integer programming approach for supply chain disruption response with lead-time awareness,” 2023.

[113] M. H. Heidary and A. Aghaie, “Risk averse sourcing in a stochastic supply chain: a simulation-optimization approach,” *Computers & Industrial Engineering*, vol. 130, pp. 62–74, 2019.

[114] A. J. Kleywegt, A. Shapiro, and T. Homem-de Mello, “The sample average approximation method for stochastic discrete optimization,” *SIAM Journal on optimization*, vol. 12, no. 2, pp. 479–502, 2002.

[115] *ISO/International Electrotechnical Commission*, ISO Std. [Online]. Available: https://iec.ch/homepage

[116] *Foundation for Intelligent Physical Agents*, IEEE Std. [Online]. Available: http://www.fipa.org/