UC Davis

UC Davis Electronic Theses and Dissertations

Title

Bayesian Networks for Real-Time Multi-Robot Task Allocation in a Generic Agent-Based Framework with Uncertainty

Permalink

https://escholarship.org/uc/item/5qz1623x

Author

Chuang, Ching-Wei

Publication Date

2023

Peer reviewed|Thesis/dissertation

eScholarship.org Powered by the California Digital Library
University of California

Bayesian Networks for Real-Time Multi-Robot Task Allocation in a Generic Agent-Based Framework with Uncertainty

By

Ching-Wei Chuang

Dissertation

Submitted in partial satisfaction of the requirements for the degree of

Doctor of Philosophy

in

Mechanical and Aerospace Engineering

in the

Office of Graduate Studies

of the

University of California

Davis

Approved:

Harry H. Cheng, Chair

Zhaodan Kong

Masakazu Soshi

Committee in Charge

2023

Copyright © 2023 by

Ching-Wei Chuang

All rights reserved.

To my parents, professors, and colleagues

Contents

List of Figures

List of Tables

Abstract

Bayesian Networks for Real-Time Multi-Robot Task Allocation in a Generic Agent-Based Framework with Uncertainty

Nowadays, robots have engaged in our daily lives no matter whether in transportation, manufacturing, military, family, education, or entertainment. Because a variety of robots have been invented in distinct environments to help human beings increase working efficiency, avoid injury in dangerous surroundings, or explore unreachable areas, more and more robots have been employed as a robot team to achieve the same goal instead of using a single robot. The use of multiple heterogeneous robots may enhance efficiency, robustness, flexibility, tolerance, and economic benefits now that each skillful robot may complete individual tasks or accomplish cooperative tasks simultaneously in a mission even though failure happens on certain robots or equipment. Multi-robot control has become much more significant than in the past decades. Although numerous structures to build centralized, decentralized, and distributed multi-robot systems (MRSs) and various approaches to coordination or multi-robot task allocation (MRTA) have been proposed, most of them may be unique in specified applications under assumed conditions. Integrating these techniques into different research areas is usually challenging. Even though certain structures and traditional or modern strategies could be combined to apply to additional applications, several challenges are still NP-hard, for example, dynamic environments, optimal solutions, parameter formalization, and parameterizing of robot abilities. An artificial intelligence (AI) method might solve these problems. With the increasing

speed of computer processing, machine-learning approaches have been commonly utilized in robotics while few of them were implemented in MRTA. In this dissertation, a novel approach to MRTA with Bayesian Networks (BNs) and a generic agent-based framework for MRSs are proposed to overcome the aforementioned difficulties. The BN of a task could be easily built, and the conditional probability table (CPT) of a BN could be logically established. The success rate of a task could be calculated by following Bayes' theorem, and tasks could be suitably allocated depending on the collected success rate. Agents in the generic agent-based framework are categorized into agencies. With this framework, a diversity of MRSs could be constructed, and advanced methodology could be implemented. Low-cost, educational hardware robots are exploited to demonstrate our approaches to MRS construction and MRTA in search-and-rescue missions under dynamic environments. The result of MRTA is acceptable with a reasonable setup before training, and the result of MRTA becomes near-optimal after training with a large data set. In the future, we may apply this new MRTA approach to a static environment, a homogeneous robot team, a large-scale robot team, and other types of MRTA problems to analyze extensibility. Researchers may include their state-of-the-art methods in the generic framework and combine our MRTA approach in miscellaneous applications, such as patrolling, surveillance, cleaning, warehouse systems, manufacturing, and exploration to examine flexibility.

Acknowledgments

First of all, I would like to thank my advisor, Professor Harry H. Cheng, for teaching me how to use the software in C-STEM Studio and providing financial support when my Ph.D. journey was close to the end, so I may focus on my research. I also want to thank my committee members, Professor Zhaodan Kong, Stavros G. Vougioukas, Masakazu Soshi, and Barbara Linke, for helping me develop my professional ability.

Second, I would like to thank my colleague, Binsen Qian. He provided me the background knowledge about the software of Mobile-C and the libraries in ChIDE. He gave constructive suggestions to help me solve the programming problems I encountered when working on my control system. He also guided me about the importance of publishing and reviewing research papers in academia. I might not be able to complete my programs without his help.

I also like to thank the department and school for supplying me with financial aid for over four years to overcome the pandemic. I am glad COVID-19 did not influence my Ph.D. journey.

Finally, I would like to thank my parents for unconditionally supporting me in studying abroad and helping me stay healthy by mailing supplies.

Chapter 1

Introduction

It is an instinct that human beings keep inventing new tools or modern technologies to improve our living quality. Since hundreds of years ago, humans have invented vehicles, boats, aircraft, and trains for transportation [1, 2, 3], drill presses, lathes, mills, CNC machines, 3D printing machines, and robotic arms for manufacturing [4, 5], and rovers, drones, underwater vehicles, rockets, and spacecraft for exploration [6, 7, 8, 9]. The examples of state-of-the-art technologies are shown in Fig. 1.1. In this modern world, besides improving existing technologies, we are still designing other robots to mimic animals for therapy or in ethology, such as robotic dogs, cats, birds, frogs, and lizards [10, 11, 12], and for services like humanoid robots [13, 14, 15], cleaning robots [16, 17], and family robots [18, 19] as shown in Fig. 1.2. Sometimes, several robots must be used at one time, for example, robotic soccer games and product lines [20, 21], and numerous rovers and drones are utilized to explore an area instead of a single robot to save time [22, 23, 24, 25]. Robots in a soccer game and a fleet of drones for exploration are shown in Fig. 1.3 Therefore, it is obvious that using many robots for one goal is inevitable no matter whether it is necessary or efficient. As a result, in addition to the mechanism design and control

system of an individual robot, effectively controlling multiple robots is another crucial research area in robotics.

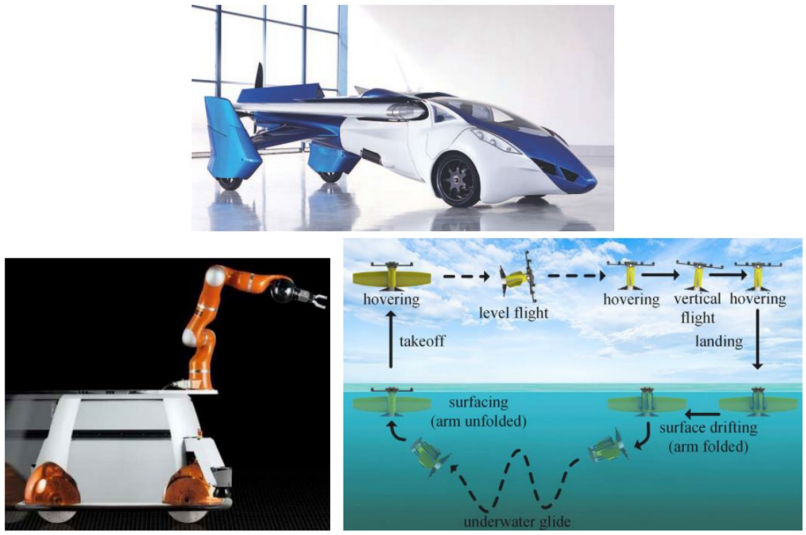

Figure 1.1. The Examples of Flying Cars, Industrial Robots, and Hybrid Aerial Underwater Vehicles [2, 5, 8]

Figure 1.2. The Examples of Cleaning Robots, Social Robots, Robotic Birds, and Humanoid Robots [10, 15, 16, 19]

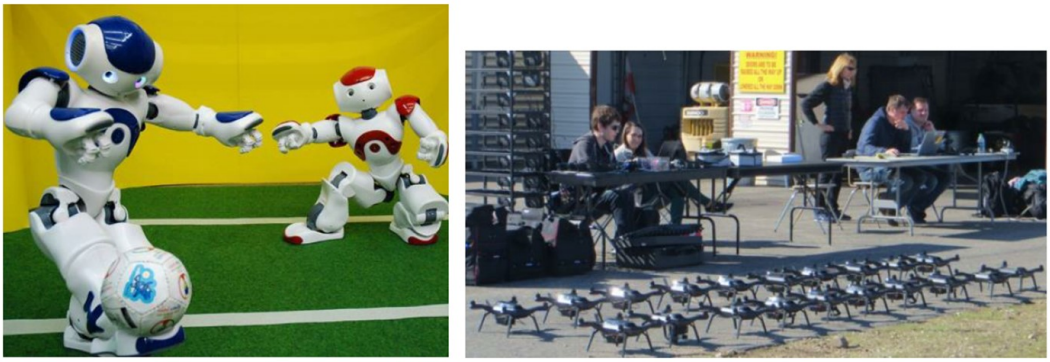

Figure 1.3. The Examples of Groups of Robots Playing Soccer and A Fleet of Drones [20, 25]

Besides the situations in which a few robots must be used, we may utilize massive robots to increase efficiency and robustness in many other applications, for instance, search and rescue, security patrolling, surveillance, material cleanup, warehouse systems, assembly systems, and exploration [26, 27]. In a small area, using one robot might be enough to achieve our purposes to search for a target or move one object from one place to another place. It is, however, very inefficient to use a single robot in a large area or to handle many jobs. Deploying various robots may save a lot of time to achieve the same goal. Additionally, increasing the number of robots may enhance robustness because robots might fail during a mission for any reason under dynamic environments. Certain functioning robots may take the jobs over from the malfunctioning robots. Moreover, the cost of each robot might decrease. When complex tasks consist of simple jobs in a mission, the requirement of a robot for each job could be lesser. The cost of a robot team would be lower because the design of each robot could be simplified [28, 29]. Consequently, with more and more applications of multiple robots, several research areas have emerged, such as system architectures, communication, information exchange and fusion, localization, coordination, resource conflict, and motion planning [30, 31, 32, 33]. Although researchers

have developed diverse approaches to coordinate robots [34], fuse information [35], localize objects [36], avoid obstacles [37], and build frameworks for a multi-robot system (MRS) [38], most of them are characteristic in specific applications. It might be uneasy to integrate these techniques into one MRS for different operations. Therefore, the main focus of this dissertation is a general design of MRS architectures and multi-robot task allocation (MRTA) which is included in coordination.

With the speed of computer calculation raising, applying machine-learning approaches to robotics becomes extremely prominent now. It might be easy to control the motion of a robot under static environments where no accidents happen and all robots follow the original schedule. Nevertheless, it might be very difficult to control a robot to react to uncertain events under dynamic environments. Take a self-driving car as an example shown in Fig. 1.4 [39]. Automatically controlling a vehicle from one place to another place in a small village with few people might be simpler than controlling a vehicle in a big city with many residents. Several situations need to be considered, such as pedestrians, traffic lights, traffic jams, and emergent accidents apart from the shortest route. With a large amount of collected data on roads every day, we might be able to properly control a vehicle after training with machine-learning approaches to design a decent self-driving car. With a similar concept, the machine-learning methodology might be able to be applied to coordination when multiple robots and dynamic factors are taken into consideration. Hence, a machine-learning approach to MRTA is proposed in this dissertation.

The structure of this dissertation is listed below. In Chapter 2, Chapter 3, and Chapter 4, multi-robot systems (MRSs), multi-robot task allocation (MRTA), and machine-

Figure 1.4. A Self-Driving Car [39]

learning approaches are reviewed separately. Motivation and research objectives are described in Chapter 5. A novel approach to MRTA with Bayesian Networks (BNs) and a generic agent-based framework are accordingly elaborated in Chapter 6 and Chapter 7. From Chapter 8 to Chapter 10, experiments with hardware robots are demonstrated in search-and-rescue missions before and after training. Conclusion and future work are discussed in Chapter 11 and Chapter 12 respectively.

Chapter 2

A Review of Multi-Robot Systems (MRSs)

A multi-robot system (MRS) consists of a group of robots that can communicate and be controlled in an area. A control system in an MRS is used to coordinate and control the motions of robots. Usually, designers develop the structure of an MRS written with certain computer languages and install it on robots to control and communicate with them. Accordingly, different types of MRSs were designed based on the structure of MRSs, the capability of robots, and the use of software frameworks in diverse applications. In this chapter, the classification of MRSs and the major software framework are introduced. The existing structures of MRSs and their advantages and disadvantages are discussed.

2.1 The Classification of Multi-Robot Systems (MRSs)

According to the capabilities robots have, MRSs can be easily divided into two types: homogeneous and heterogeneous MRSs [40]. If each robot in a group owns the same skill, we call it a homogeneous MRS; on the other hand, if robots possess different skills in a

group, we name it a heterogeneous MRS. In a dynamic environment, even though robots in a team have the same capability, the capability conditions on each robot might be different like broken sensors or failing actuators, so we may categorize this robot team into a heterogeneous MRS.

Based on coordination methods, MRSs can be classified as centralized, decentralized, and distributed MRSs [26, 41] as shown in Fig. 2.1. A centralized MRS is a system where only one fixed central controller exists. This controller can control all robots including itself when the controller is one of the robots. The central controller collects information from robots, analyzes current conditions, coordinates robots, and controls robots. Other robots only follow commands from the central controller to complete what needs to be done. The strength of this system is that it is simple for the central controller to make decisions and provide proper commands to robots by analyzing current situations with certain algorithms because all information is collected by the central controller. However, the weakness is that the system may not work when the central controller fails. Without the central controller, robots do not know what to do and stop moving, which is so-called a single-point failure [31, 42]. Moreover, the cost of the central controller is typically

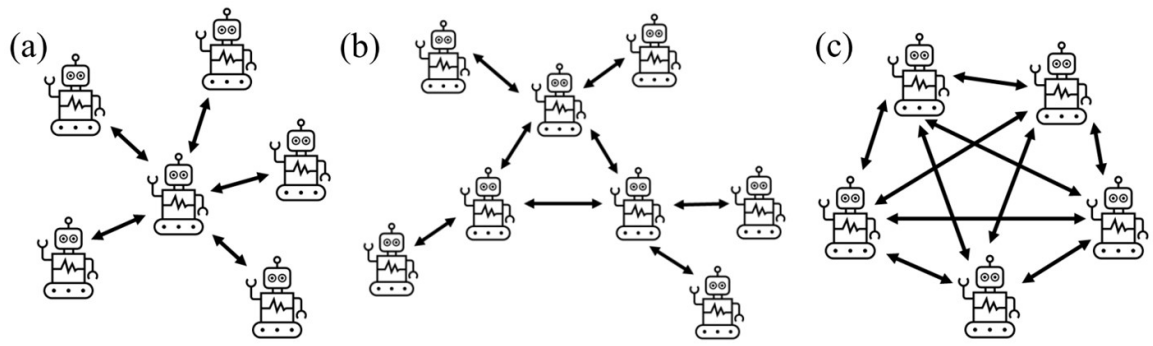

Figure 2.1. The Classification of MRSs: (a) Centralized (b) Decentralized (c) Distributed

expensive since it has to handle heavy calculations and analyses, especially in a large-scale robot team, so a high-performance computing processor might be necessary. To overcome single-point failures, decentralized and distributed MRSs were developed.

Many settled or transient controllers are in a decentralized MRS. Certain assigned robots may be controllers to gather and fuse information or coordinate and control robots including themselves. Heavier duties can be done on high-efficiency robots, and lighter work can be done on low-efficiency robots. A decentralized MRS may still work although the robots responsible for coordination fail, and the cost of a robot might decrease due to the nonessential use of high-performance computing processors, which solves the problems in a centralized MRS. Nevertheless, the requirement for communication between robots is very critical [43]. A robot might wrongly analyze current situations with out-of-date or incorrect information because of delay and send erroneous commands to robots, which might cause robots to conduct improper behaviors. Thus, a distributed MRS was proposed.

Each robot in a distributed MRS can exchange information with other robots, make decisions depending on local and gathered information, and control itself. Even though a robot loses communication, it can still try to collect information by sensing its surroundings and to analyze current situations. Consequently, a group of robots may complete what operators require to do although certain robots fail. Nonetheless, information fusion is relatively challenging compared to the other two MRSs for the reason that information is collected at different locations or at different points in time. Various complex algorithms or approaches to information fusion are required under distinctive conditions [44, 45, 46].

	Structures	Heterogeneity	Environments
Search and Rescue [47, 48, 49, 50]	centralized/ decentralized/ distributed	heterogeneous	dynamic
Security Patrolling [51, 52, 53, 54, 55]	centralized/ distributed	homogeneous/ heterogeneous	static
Surveillance [56, 57, 58]	centralized/ decentralized	homogeneous	static
Material Cleanup [59, 60, 61]	centralized/ distributed	heterogeneous	dynamic
Warehouse Systems [62, 63]	centralized/ decentralized	homogeneous	static
Assembly Systems [64, 65]	centralized	heterogeneous	dynamic
Exploration [66, 67, 68]	centralized/ distributed	homogeneous/ heterogeneous	dynamic

Table 2.1. The Complexity in Different Applications

Because there are pros and cons in centralized, decentralized, and distributed MRSs, we should think thoroughly before choosing a suitable MRS for an application. The typical applications of MRSs are organized in Table 2.1. Centralized MRSs are commonly used in most applications if the size of a robot team is small or the environment is relatively static because the central controller can easily analyze collected information and control robots, for example in security patrolling, surveillance, warehouse systems, and assembly systems. When a robot team is large or the environment is dynamic, decentralized or distributed MRSs are utilized to quickly gather surrounding information and react to accidents, such as in search and rescue, material cleanup, and exploration. Regarding heterogeneity,

robots with the same skills are frequently used in security patrolling, surveillance, and warehouse systems, whereas robots with different capabilities are generally utilized in search and rescue and material cleanup. As can be seen, types of MRSs and robot styles vary in diverse applications, so our proposed approach should be applicable to most applications.

2.2 The Existing Software Framework of Multi-Robot Systems (MRSs)

In order to create centralized, decentralized, or distributed MRSs, several existing MRS software frameworks were developed. The software frameworks of MRSs are the platforms with which researchers may build personalized MRSs with certain computer languages depending on their applications to monitor and control robots. The existing MRS software frameworks are OpenRTM, OROCOS, YARP, ROS, JADE, and Mobile-C [69]. They all have their characteristics compared to each other in several areas; notwithstanding, various factors should be taken into consideration when choosing a software framework to create an MRS. For instance, most software frameworks are compatible with C/C++, whereas Java is used in JADE. Using C/C++ to control devices, such as sensors, motors, and cameras, is very common because C/C++ is a compiled language, the processing speed of which is faster than the speed of interpreted languages. For another example, robots may need to keep perceiving surroundings from sensors or cameras in dynamic environments, so real-time control is crucial. OROCOS, YARP, and Mobile-C can achieve this purpose. Last, JADE and Mobile-C are mobile agent-based software frameworks, so certain agents are able to move to other devices to run certain programs and bring the

required data back. Considering that our proposed approach should be handily utilized in diverse applications, Mobile-C is chosen as our software framework in this dissertation to build MRSs on hardware robots.

Mobile-C is written in C/C++ and follows the standard of the Foundation for Intelligent Physical Agents (FIPA), so it is easy to control hardware and send messages with XML files [70, 71]. The basic structure of Mobile-C is shown in Fig. 2.2. Each robot may have several agencies as servers or clients to send and receive messages. In each agency, we may create numerous stationary agents (SA) or mobile agents (MA) to execute diverse tasks. The difference between stationary agents and mobile agents is that stationary agents can only execute tasks locally and mobile agents are able to go to other agencies to execute tasks on different robots via the network. Through the agent communication channel (ACC), agents are allowed to send and receive messages or data from each other by using an agent communication language (ACL). Additionally, agents

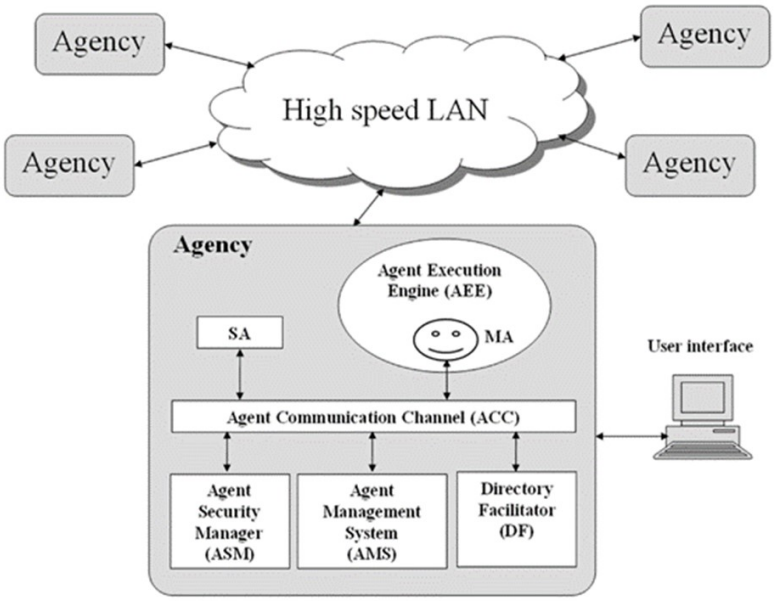

Figure 2.2. The Structure of Mobile-C [70]

may communicate with each other asynchronously because they are able to re-transmit information if a receiver temporarily fails. Users may also create mobile agents in one local agency and send them to other agencies on different robots to obtain the necessary information or to manually control robots. As a result, Mobile-C has several advantages, such as flexibility, mobility, communication, and robustness.

2.3 The Existent Structure of Multi-Robot Systems (MRSs)

Due to various types of MRSs and applications, numerous MRSs have been developed, whereas an agent-based architecture has been seen as a feasible solution to build an MRS because of its flexibility, scalability, distribution, extensibility, modularity, robust fault tolerance, and real-time control [72]. In [73], although researchers introduced a fundamental agent-based framework for an MRS, implementation and the details about agents' names and their work were not provided. In [74, 75], scholars listed sensor agents, actuator agents, communication agents, planner agents, mapping agents, and localization agents and showed the dataflow between agents. Certain agents could save data in the knowledge base (KB) or the blackboard. However, they did not consider any agents handling coordination between robots. In [72], researchers classified agents from another angle. Master agents are responsible for high-level coordination, server agents deal with low-level planning, and worker agents handle hardware control. Communication agents and task scheduling agents are involved in these classified agents. Nevertheless, they did not take information fusion and diverse task-allocation approaches into account although considering additional tasks and robots during a mission. In [76], research workers

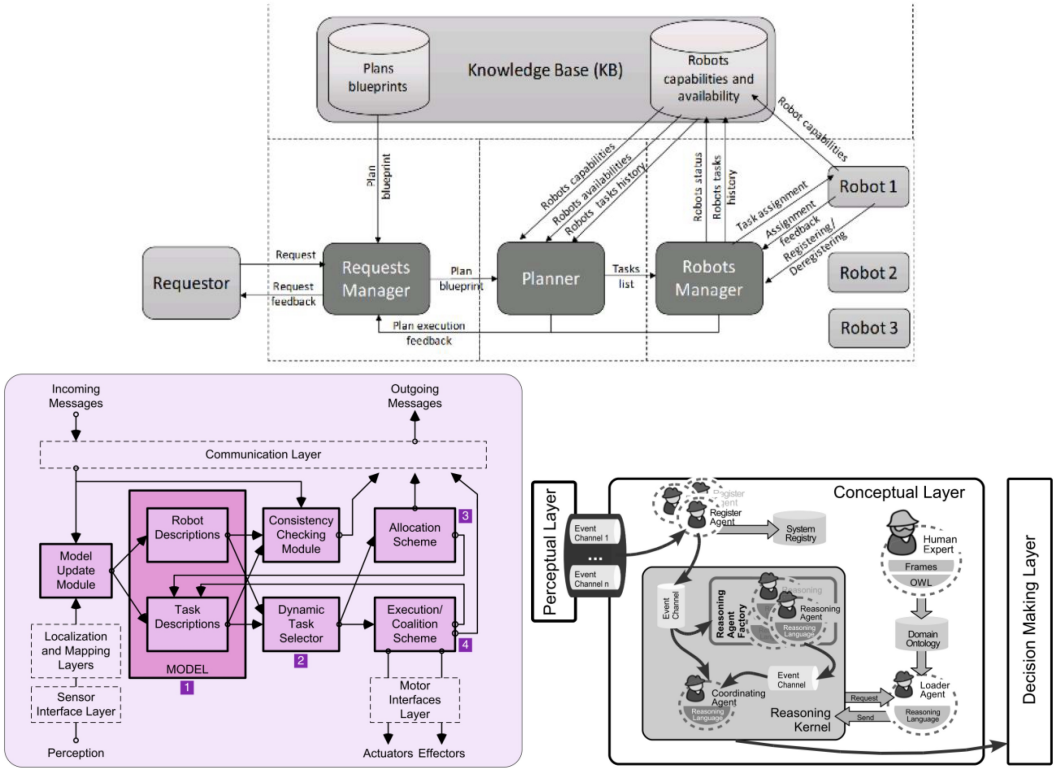

Figure 2.3. The Examples of The Present MRS Structures [76, 79, 84]

mainly focused on coordination; hence, apart from sensor agents, actuator agents, mapping agents, localization agents, and communication agents, they included several agents to manage task allocation and coalition formation in the coordination module. Due to this reason, they did not mention agents to handle multi-sensor fusion or object recognition. Up to now, several researchers have proposed various frameworks or architectures with different types of MRSs and approaches in numerous software and applications [77, 78, 79, 80, 81, 82, 83, 84]. Multiple examples are shown in Fig. 2.3. Although we are able to point out certain crucial agents and features in agent-based frameworks, most of them are unique in specific algorithms, software, or applications. As a result, proposing a generic agent-based framework is crucial to apply to most applications without massive changes.

Chapter 3

A Review of Multi-Robot Task Allocation (MRTA)

In autonomous MRSs, one primary research area is coordination including task decomposition, coalition formation, task allocation, and motion coordination [85]. When a mission is assigned to an autonomous robot team, task decomposition is the first step to separate the mission into several simple tasks for robots. Nonetheless, utilizing certain algorithms or approaches to automatically decompose a mission is a tough problem. The task category varies case by case, so the task list of a mission is manually created and given by researchers in advance [86, 87, 88]. Thereafter, certain approaches are applied to effectively assign tasks to robots, which is called task allocation, and small groups of robots may be formed in collaborative tasks, which is named coalition formation. In the end, robots may coordinate among them to complete assigned tasks with customized methods. In this dissertation, one focus is task allocation including coalition formation because the result of task allocation may directly influence the execution outcome. The classification of MRTA, current approaches to MRTA, and challenges are concluded in this chapter.

3.1 The Classification of Multi-Robot Task Allocation (MRTA)

MRTA is one of the research areas in coordination to appropriately allocate tasks to robots in a mission in order to properly complete the mission within limited resources. In accordance with research papers related to MRTA, the problems of MRTA could be classified into eight types based on the type of robot, the requirement of tasks, and assignment methods as shown in Fig. 3.1 [89, 90]. Robots in a team could be categorized into single-task (ST) robots and multi-task (MT) robots. An ST robot can only execute one task at a time. It must complete or give up its current task before taking another task. Contrarily, an MT robot can simultaneously execute many tasks. It may not need to complete or abandon certain current tasks before taking others. With respect to the requirement of tasks, tasks could be divided into single-robot (SR) tasks and multi-robot (MR) tasks. Only one robot is required in an SR task, whereas multiple robots are needed in an MR task. The number of robots and the required ability of robots may vary in each

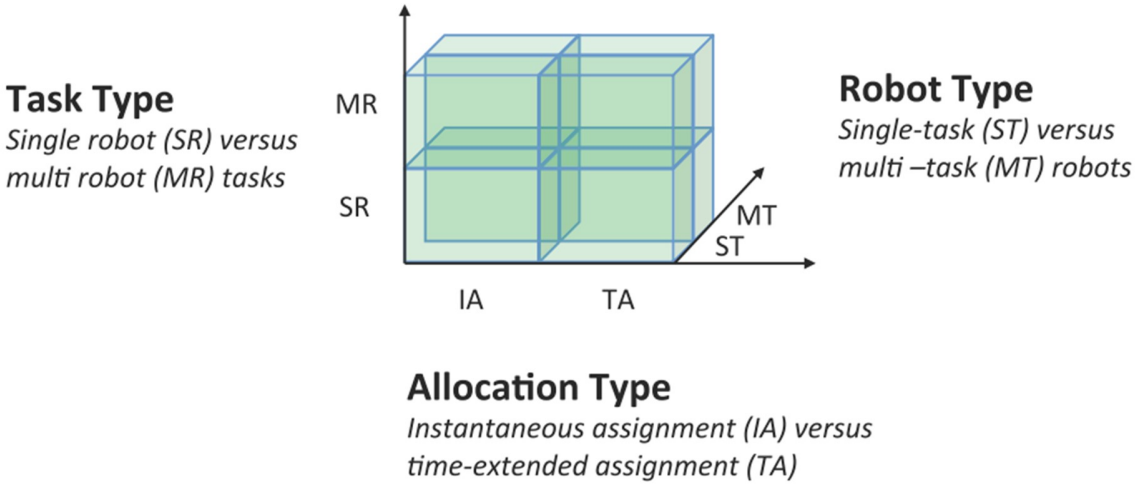

Figure 3.1. The Classification of MRTA Problems [89]

task. According to the information between tasks, we classify assignment methods into instantaneous assignments (IA) and time-extended assignments (TA). IA means there is no sequential information between tasks, and tasks are executed immediately when being taken by robots. TA denotes tasks are assigned by considering the sequential information of tasks, and some tasks may be executed later although being chosen by robots. Hence, eight combinations are in MRTA problems: ST-SR-IA, ST-SR-TA, ST-MR-IA, ST-MR-TA, MT-SR-IA, MT-SR-TA, MT-MR-IA, and MT-MR-TA.

3.2 The Current Approach to Multi-Robot Task Allocation (MRTA)

Up to now, several researchers have proposed numerous approaches to solve different types of MRTA problems under miscellaneous conditions [90, 91, 92, 93]. The three fundamental approaches are behavior-based approaches, optimization-based approaches, and auction-based approaches. Other strategies which are categorized as hybrid approaches may combine distinctive methods with them.

In behavior-based approaches, designers build a structure to store several behaviors, roles, or tasks on each robot. Each robot may choose one of them based on local information collected through sensors and external information assembled from other robots. This method is usually used in a heterogeneous, distributed MRS because the structure installed in a robot could be the same or slightly different based on applications and each robot may make its own decisions according to gathered information. Additionally, this method may adapt to dynamic environments now that robots may easily choose or switch to different behaviors, roles, or tasks depending on combined information. The classic

examples are ALLIANCE [94] and L-ALLIANCE, which is a machine-learning version of ALLIANCE [95]. However, it is rigorous to design the structures of behavior-based approaches [73, 90]. Researchers need to build different structures when diverse behaviors, roles, or tasks are required in various applications. Moreover, there is no general algorithm to determine how a robot chooses suitable behaviors, roles, or tasks in a mission due to distinct situations. Thus, other approaches were proposed.

In optimization-based approaches, a central controller or a robot collects all information from other robots and utilizes an optimizer to minimize the cost based on objective functions with constraints in order to obtain the global, optimal solution in MRTA. The common optimizers used in MRTA are Ant Colony Optimization (AC) [96], Particle Swarm Optimization (PSO) [97], Genetic Algorithm (GA) [98], and Simulated Annealing (SA) [99]. This method is basically utilized in a homogeneous, centralized MRS because all robots are assumed to be the same and a central controller should collect and analyze information and then send commands to robots. Furthermore, the global, optimal solution, with this method, can be easily obtained after collecting enough information about the surroundings. Nevertheless, due to a centralized MRS, weaknesses are accompanied, such as a single-point failure and high-cost hardware. In addition, many objective functions or different algorithms may be needed or re-designed for complicated applications, so objective functions vary in diverse situations. Hence, auction-based approaches emerged.

In auction-based approaches, three main stages in sequence are in an auction process: Announcement Stage, Submission Stage, and Selection Stage [100]. First, in the Announcement Stage, a robot selected by a designer or a group of robots as an auctioneer

announces a task to all robots. Next, in the Submission Stage, after a robot receives the task, it calculates a utility as a bid with certain mathematical algorithms based on its current conditions, such as the cost and the reward to complete the task. Then, it will send the bid back to the auctioneer. Finally, in the Selection Stage, after the auctioneer collects all the bids from the robots, it will select the winner with the highest bid and send the task to the robot. After the auction process completes, an auctioneer starts another auction process and repeats all the stages continually until all tasks are allocated. The overall process is shown in Fig. 3.2. Based on the information about the auction process, this method is usually utilized in a heterogeneous, decentralized/distributed MRS and overcomes certain challenges, for example, the single-point failure, the structure design, and dynamic environments [101]. A simple example is shown in [60] where researchers utilized an auction-based approach to allocate cleaning tasks to robots with different abilities after the robots observed their surroundings. Researchers in [102, 103] also stated the efficiency, robustness, and scalability of auction-based approaches. Nonetheless, applying auction-based approaches may receive a sub-optimal solution instead of the optimal solution [104]. In order to obtain better results, certain researchers proposed modified

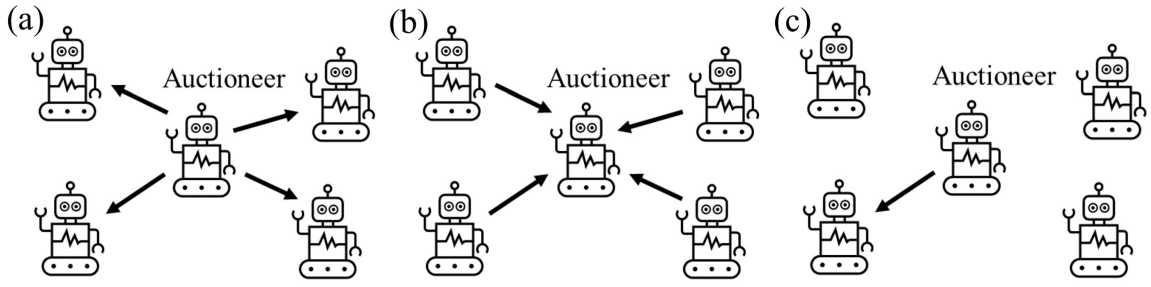

Figure 3.2. An Auction-Based Approach: (a) Announcement Stage. (b) Submission Stage. (c) Selection Stage.

auction-based approaches, such as the greedy auction algorithm [105] and the market-based approach [106]. Moreover, in order to decrease the occasions of auction processes, researchers utilized the combinatorial exchange to bundle tasks or robots as a group or a party in auction processes [107]. Other cutting-edge strategies were illustrated and compared in [108]. As can be seen, simple auction-based approaches might not be applicable to receive the optimal result, whereas modified auction-based approaches might heavily increase the difficulty of auction processes. Hybrid approaches were developed.

Hybrid approaches may include adding distinctive algorithms to the fundamental approaches (behavior-based, optimization-based, and auction-based approaches) or combining two of the fundamental approaches. For instance, a consensus algorithm was added in an auction-based approach to create a consensus-based approach in [109, 110] to receive better results in a heterogeneous MRS than the result with simple auction-based approaches. For another example, an optimization-based approach was combined with an auction-based approach, which is called the stochastic clustering auction, to obtain the optimal solution [111]. Today, certain challenges are still unsolved although several techniques have been proposed.

3.3 Challenges and Difficulties in Multi-Robot Task Allocation (MRTA)

We may notice that certain difficulties and challenges still exist based on the descriptions in the previous section even though miscellaneous approaches have been proposed. Most structures, parameters, and algorithms are empirically customized according to distinctive applications. The structure design to store and choose tasks would be different

and troublesome with behavior-based approaches. Objective functions and constraints would be various with optimization-based approaches. Costs, rewards, and the calculation of utilities would be diverse with auction-based approaches. Extra algorithms may be added in auction-based approaches to receive near-optimal solutions, which increases the calculation complexity. Additionally, parameters used in most methodologies may be created by trial and error. Algorithms may become extremely complicated to obtain the optimal solution or overcome dynamic environments including adjustable task lists. For example, sophisticated hybrid approaches were proposed to handle dynamic conditions in [112, 113, 114]. The robot cost might augment due to the hardware requirement for heavy calculation. Most importantly, it is rare to formalize the capability of a robot in an algorithm although the ability condition was considered in the specific case in [115]. Last, most approaches are not learnable to adapt to dynamic environments or new missions. The original setup might not work when the information about robots and tasks changes. As a result of the above challenges and difficulties, a novel approach to MRTA with a machine-learning method is proposed in this dissertation.

Chapter 4

Introduction to Machine-Learning Approaches in Robotics

With the computing speed of computer boosting, machine-learning approaches become prominent to be utilized in education, human resource, cybersecurity, health care, military, manufacturing, and robotics [116]. Because of the complexity of mechanisms increasing, more and more machine-learning approaches are implemented in robotic research areas because deriving a deliberative method might be too complicated and problematic due to many inputs and outputs, dynamic factors, and uncertainties. After training with a large, collected data set, a trained machine-learning algorithm may provide a feasible solution to allow robots to analyze surroundings, make decisions, and choose behaviors. In this chapter, a summary of current machine-learning approaches in robotics is introduced. The reason that Bayesian Networks (BNs) might be a decent option for MRTA is elaborated. In the end, a brief introduction to BNs is given.

4.1 The Category of Machine-Learning Approaches and Applications in Robotics

By now, several machine-learning approaches in Artificial Intelligence (AI) have been developed to allow machines to learn from data. Most methods can be categorized as supervised learning, unsupervised learning, semi-supervised learning, reinforcement learning (RL), and ensemble learning as shown in Fig. 4.1 [117, 118, 119, 120]. Supervised learning means the collected data for training are labeled while unlabeled data are used in unsupervised learning. RL is a learning process when only goals are given in Markov Decision Process (MDP) or Partially Observable Markov Decision Process (POMDP). Ensemble learning is a learning process where multiple learning models are utilized to solve one common problem. Approaches in supervised learning are Decision Tree, Naive Bayes and Bayesian Networks (BNs) or Belief Networks, Support Vector Machines (SVM), K-nearest Neighbors (K-NN), and linear regression. Unsupervised-learning approaches are K-Means and Principal Component Analysis. Classical methods in ensemble learning are Random

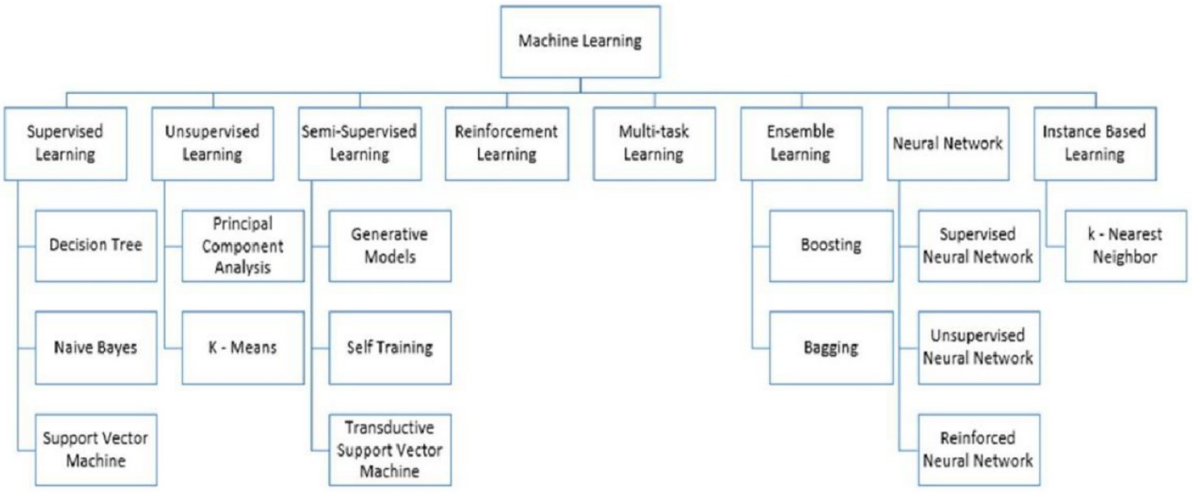

Figure 4.1. The Categorization of Machine-Learning Approaches [117]

Forest and Boosting. The well-known approach, Neural Networks (NN), can be utilized in supervised learning, unsupervised learning, and reinforcement learning. Another remarkable approach is deep learning (DL) where multifold layers are used in NN, so it can be combined with NN, BNs, and RL to solve much more complex problems [121].

During the past decades, numerous machine-learning approaches have been applied to robotics in many research areas, such as object recognition, data fusion, localization, coordination, path planning, collision avoidance, grasping, and fault detection. Each classification or clustering approach like SVM and BNs may be utilized to detect and recognize objects, whereas it is very powerful to identify targets in a messy environment with NN or DL [122, 123, 124]. Data fusion may be required to extract and integrate crucial features from one source or multiple sources like multi-sensor fusion to estimate the current state or analyze surroundings for further decision making. Boosting, SVM, BNs, K-Means, NN, and DL are general approaches in data fusion [125, 126, 127, 128]. Localization problems may include indoor and outdoor self-localization and target localization. Because these problems are usually related to images and sensors, certain approaches utilized in object recognition and data fusion may be needed. Additionally, current pose estimation may be also required, so POMDP, BNs, linear regression, SVM, K-NN, and NN have been applied to localization [129, 130, 131, 132]. Coordination is essential in an MRS to allocate tasks, share limited spaces, make robots travel as a group, or collaboratively complete a task. Due to dynamic environments, different communication methods, and uncertainties, RL and DL are widely utilized to handle sophisticated problems [133, 134, 135, 136]. In MRTA, however, the simulation results in particular cases are under specific assump-

tions and without robot failures during execution although learning-based strategies were summarized in [137]. The trained model might be unable to handle authentic situations when task information and robot capabilities change with time. Re-training might be required. Path planning and collision avoidance occur when a robot executes a task and tries to avoid any static or dynamic obstacles on a road. Because of many uncertain situations on a path, BNs, NN, RL, and DL are implemented to handle any possible circumstances [138, 139, 140, 141]. Grasping and relocating an object can be a task in a mission. Object recognition sometimes accompanies grasping tasks; hence, BNs, NN, RL, and DL are adopted to control robotic arms or grippers to capture rigid or soft items [142, 143, 144, 145, 146]. Fault detection is the process of data analysis to determine the status of hardware; therefore, certain approaches to data fusion may be included. Boosting, linear regression, SVM, K-NN, and NN are typical methods used in fault detection [147, 148, 149]. According to these applications and characteristics of machine-learning approaches, as a result, we may choose one of them to solve a diversity of MRTA problems and consider as many possibilities as possible when designing a generic MRS framework.

4.2 Promising Machine-learning Approaches in Multi-Robot Task Allocation (MRTA)

Although machine-learning approaches are very popular in robotics, it is rare to apply a machine-learning method to MRTA while considering several applications. In consequence, our goal is to choose one of them to overcome the challenges described in Chapter 3. Based on the characteristics of MRTA problems, not all of them can be fulfilled. For example, many dynamic factors need to be considered, such as the state of a robot, the

number of tasks, the information of a task, the type of robot, the skill condition, and uncertainties from devices. Decision Tree, Naive Bayes, BNs, NN, and RL including MDP and POMDP might be appropriate approaches to handle dynamic environments [150, 151]. Nonetheless, considering the difficulty of building a structure to solve MRTA problems, Decision Tree might not be a proper approach because the structure might be totally different when the number of tasks, the number of robots, or the information of tasks varies in diverse applications. Furthermore, although MDP or POMDP is powerful to obtain the optimal sequence of actions to complete a task, an MRTA problem, especially for an IA problem, is similar to a classification problem under dynamic environments. Finding an optimal sequence of task execution is trivial since it is almost impossible to estimate what will happen to robots, sensors, or tasks. MDP or POMDP might not be a satisfactory approach in classification [85, 152]. Lastly, the difficulty of learning should be taken into account. In real applications, tasks, robots, and surroundings are miscellaneous. Training the model of a machine-learning approach might require a huge data set, whereas it is difficult to collect a large amount of data from real experiments. Meanwhile, the modification of a structure or re-training might be needed in various applications. NN or RL might not be applicable with limited data for learning [151, 153, 154]. Consequently, BNs or Naive Bayes, which is a branch of BNs, might be adequate for MRTA.

4.3 Introduction to Bayesian Networks (BNs)

A Bayesian Network (BN) is a model-based approach in machine learning [155] and is useful in reasoning and inference with uncertainties [150]. The structure of a BN is a directed acyclic graph (DAG), which means there is no closed chain in a framework

compared to a Markov chain in MDP [118]. There are three fundamental steps in BNs: constructing the structure of a BN, creating conditional probability tables (CPTs) for each node, and applying Bayes' theorem to calculate a posterior probability [156]. A very simple example of a BN is shown in Fig. 4.2. There are two nodes in this BN: Node A and Node B. The arrow means a direct causal influence. In this example, Node A directly influences Node B, so Node A is a parent of Node B, or Node B is a descendant of Node A. Thereafter, the CPTs of Node A and Node B can be created. Examples are shown in Table 4.1 and Table 4.2. Node A has two values, Yes and No, and Node B also has two values, Yes and No. The conditional probabilities in the tables are generated based on facts or a massive amount of data. In the end, Bayes' theorem is used to calculate a posterior probability. The definition of Bayes' theorem is in Eq. (4.1). With this equation, we may calculate the posterior probability we are interested in. For example, the posterior probability P(A|B) can be calculated when A is Yes and B is Yes, and the result is in Eq. (4.2). More details will be given in Chapter 6 when elaborating on our approach to MRTA.

Figure 4.2. An Example of a Bayesian Network

A	P(A)
Yes	0.5
No	0.5

Table 4.1. The CPT of Node A

A	B	P(B|A)
Yes	Yes	0.7
Yes	No	0.3
No	Yes	0.2
No	No	0.8

Table 4.2. The CPT of Node B

$$P(A|B) = \frac{P(A,B)}{P(B)} = \frac{P(B|A)P(A)}{P(B)}, \ \ where\ P(B) = \sum_{\forall a \in A} P(B|a)P(a) \tag{4.1}$$

$$P(A|B) = \frac{P(A,B)}{P(B)} = \frac{P(B|A)P(A)}{P(B)} = \frac{0.7 \times 0.5}{0.7 \times 0.5 + 0.2 \times 0.5} = 0.7778 \tag{4.2}$$

Chapter 5

Motivation and Research Objectives

After reviewing the current MRSs, they could be classified as centralized, decentralized, and distributed regarding the communication type or homogeneous and heterogeneous depending on the robot style in numerous applications. Each structure in MRS research papers might be one of a kind based on their proposed approaches to specific problems, so other researchers might need to re-design totally different structures in order to apply their algorithms to MRSs. Besides diverse MRS frameworks, approaches to MRTA are various according to eight types of MRTA problems and static or dynamic environments, which might also cause different MRS structures to be needed. Even though three main approaches (behavior-based, optimization-based, and auction-based approaches) to MRTA were proposed, certain challenges or difficulties are still unsolved, for example, design complexity, numerical parameterization of skills or abilities, formalization of parameters, calculation complexity, dynamic environments including changeable task lists, the optimal solution, and learning. According to the information in Chapter 3, the structures of behavior-based approaches are hard to design although this method is learnable. The calculation could be very complicated to obtain the optimal solution with many

objective functions and constraints in optimization-based approaches. Auction-based approaches are widely utilized to overcome dynamic environments while the result may be sub-optimal and costs and rewards are set by trial and error. Notwithstanding, in recent literature about MRTA, the condition of robot capabilities was rarely discussed, and the parameterization of robot skills has not been analyzed. As a result, we are encouraged to propose a novel approach to MRTA to overcome these challenges and a generic MRS framework that could be applied to miscellaneous applications without massive re-design. The research objectives are listed below.

- Only instantaneous-assignment (IA) problems in MRTA are considered due to dynamic environments.
- A novel approach with BNs is proposed to solve existing problems in MRTA.
- The BN of a task should be easily built and should combine crucial dynamic factors including robot states, skill conditions, and task requirements.
- The CPTs of a BN could be reasonably and logically designed.
- The success rate of a task is considered when a robot executes the task.
- The calculation to obtain the optimal solution should be simple in the MRTA process.
- The proposed approach to MRTA is learnable and can be used in various MRSs.
- The generic framework to build MRSs is proposed for diverse applications.

- The generic agent-based framework could be applied with new techniques without substantial re-design.

- The proposed approach would be demonstrated with low-cost hardware robots to solve different types of MRTA problems in a variety of MRSs.

- The proposed approach is applied to a realistic search-and-rescue mission to illustrate how this methodology deals with dynamic environments.

- The MRTA result in the mission would be acceptable with untrained CPTs and near-optimal with trained CPTs.

Chapter 6

The Novel Approach with Bayesian Networks in Multi-Robot Task Allocation (MRTA)

In this chapter, detailed information about the novel approach to MRTA with Bayesian Networks (BNs) is elaborated. The general structure of the Bayesian Network of a task is introduced in the beginning, and the initial conditional probability tables (CPTs) in each task are reasonably constructed based on experts' knowledge [157]. Besides building initial CPTs with rational assumptions, the CPTs can be trained or updated with a large amount of data to obtain the near-optimal result. The crucial concept of this MRTA approach is to allocate tasks according to the values of success rates calculated from BNs. Algorithms and rules are established to accommodate most MRTA applications. In the end, the comparison of MRTA approaches is analyzed.

6.1 The General Bayesian Network of a Task

Before introducing the general Bayesian Network of a task, we should understand what information tasks contain in a mission. A mission is usually composed of many small tasks.

For instance, in a search-and-rescue mission, a task requires a robot to search for a target in an area, and another task asks robots to move a target to a safe place. In a surveillance mission, a task needs a robot to survey at a point, and another task requests a robot to monitor surroundings at another point. In a warehouse, a robot needs to transport goods from place to place, which can represent many tasks in a transportation mission. As can be seen, most missions can be separated into several independent, small tasks. Generally, a designer or a researcher creates the task list of a mission before directly assigning the mission to a robot team because it is very difficult to automatically decompose a mission into many applicable tasks with certain algorithms for a robot team. An independent task designed by an operator in a mission, therefore, may include the number of required robots, the demanded capability robots should have, target locations, and the job to be completed.

According to the current challenges in MRTA in Chapter 3, many dynamic factors need to be considered, such as the information of a task, the information of a robot, the information of ability, and the cost to complete a task. However, a formalized approach to combining these factors is rare. A Bayesian Network, hence, is utilized to calculate the success rate of a task in the proposed approach in order to invent a generic MRTA method. The nodes and causal relationships in the general Bayesian Network of a task are listed below, and the general Bayesian Network of a task is shown in Fig. 6.1.

Nodes:

- The Condition of a Robot (Node A and Node E): There are two nodes to represent the current conditions of a robot: the energy condition and the conditions of current,

taken tasks. The conditions of current, taken tasks may influence whether a robot is suitable to take another task. The energy condition may affect whether a robot could successfully complete the task.

- The Type of Robot (Node B): This is a node to set up whether a robot is a single-task (ST) or multi-task (MT) robot.
- Task (Node C): This node represents if a robot takes the task or not and is used to calculate the posterior probability (the success rate).
- Cost (Node D): This node shows the estimated value of the cost to complete the task. It can be represented in energy, time, or distance [158].

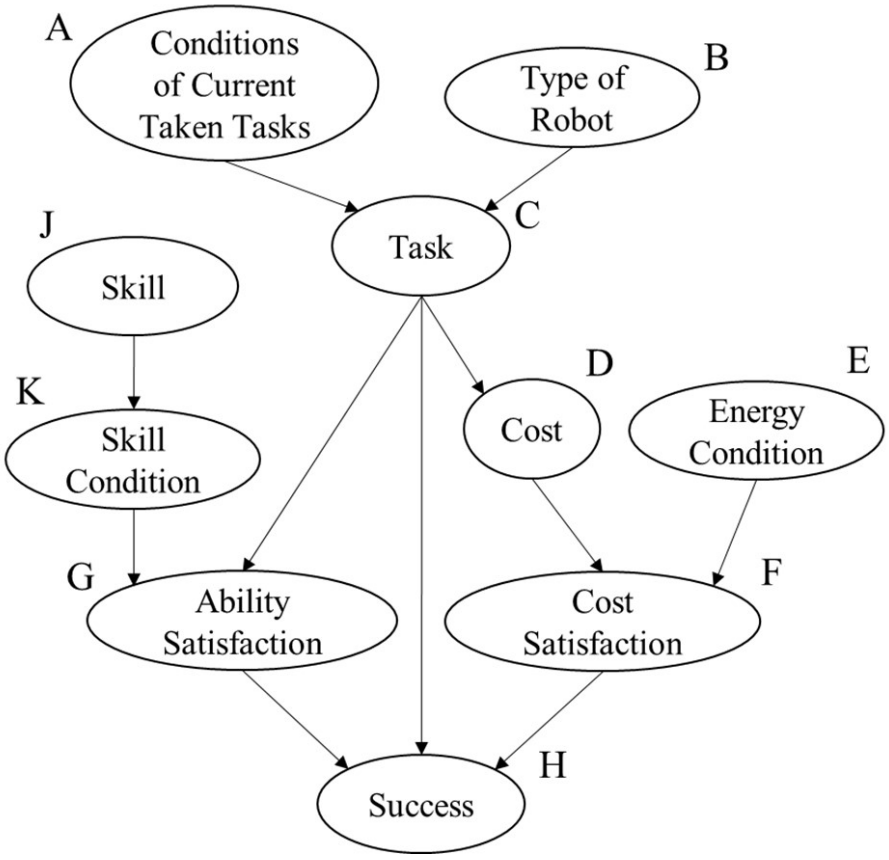

Figure 6.1. The General Bayesian Network of a Task

- Skill (Node J): This is a node to determine whether a robot has a skill, such as driving, flying, sensors, cameras, and arms. Each skill node represents one skill, so it is possible to have many skill nodes in one BN of a task.

- Skill Condition (Node K): This node is used to represent the skill condition because the condition (the quality or the accuracy) varies with time and diverse hardware. The number of this node is the same as the number of the skill.

- Cost Satisfaction (Node F): It is a node to conclude whether the energy condition on a robot is good enough to complete the task.

- Ability Satisfaction (Node G): It is a node to determine whether the robot skills are satisfied with the task requirements.

- Success (Node H): This node is used to decide whether a robot successfully completes the task and to calculate the posterior probability (the success rate).

Causal Relationships Between Nodes:

- The conditions of current, taken tasks (Node A) and the type of robot (Node B) may affect the probability that a robot takes the task (Node C).

- A task (Node C) must contain certain requirements (required skills) and a cost (Node D) which can be calculated by referring to the information of the task and a robot.

- One Skill Node (Node J) must be accompanied by the Skill Condition Node (Node K).

- The cost (Node D) and the energy condition (Node E) may influence the result of the Cost Satisfaction Node (Node F).

- The requirement of the task (Node C) and the skill condition (Node K) may affect the result of the Ability Satisfaction Node (Node G).

- The results of the Cost Satisfaction Node (Node F) and the Ability Satisfaction Node (Node G) may conclude whether a robot can successfully complete the task (Node H).

6.2 The Creation of Conditional Probability Tables (CPTs)

After building the BN of a task, the probabilities of Nodes A, B, E, and J can be calculated, and the CPTs of Nodes C, D, F, G, H, and K can be created. In order to receive a result that is near to real situations, we are supposed to collect data from experiments or real missions and then generate CPTs in the BN because a BN is one of the model-based approaches in artificial intelligence (AI). Nonetheless, we may also design CPTs first based on reasonable assumptions with experts' knowledge and then update them, which is a so-called learning process in machine learning [159]. Consequently, the initial probabilities and the initial CPTs are assumed and schemed in this dissertation as follows for experiments and case studies.

- Node A (Table 6.1): This node is assumed to represent the number of current, taken tasks in other missions. It has four values: 0, 1, 2, and more than 2. The probability distribution is uniform because the number of tasks a robot currently takes varies

with situations and time.

- Node B (Table 6.2): There are two values in this node: ST and MT. The probability can be calculated according to the number of ST and MT robots in a team. One important note is that the value of a probability cannot be zero or one but with the decimal value which is very close to zero or one instead [156]. More information can be found in comprehensive experiments.

- Node C (Table 6.3): The values in this node are Yes and No to decide whether a robot takes the task. If an ST robot has already had a task in other missions, the conditional probability to take the task in the mission should be very small (close to zero). On the other hand, the conditional probability to take the task in the mission should be smaller when an MT robot takes more tasks in other missions in order to distribute the task to other robots.

A	**P(A)**
0	0.250
1	0.250
2	0.250
Larger than 2	0.250

Table 6.1. The Probability Table of Node A

B	**P(B)**
ST	0.999
MT	0.001

Table 6.2. The Probability Table of Node B (All Robots Are ST)

P(C=Yes|AB)		B	
		ST	MT
A	0	0.900	0.900
	1	0.100	0.700
	2	0.010	0.500
	More than 2	0.001	0.300

Table 6.3. The CPT of Node C

D	P(D|C=Yes)	P(D|C=No)
0∼3 sec	0.066667	0.066667
3∼6 sec	0.066667	0.066667
6∼9 sec	0.066667	0.066667
9∼12 sec	0.066667	0.066667
12∼15 sec	0.066667	0.066667
15∼18 sec	0.066667	0.066667
18∼21 sec	0.066667	0.066667
21∼24 sec	0.066667	0.066667
24∼27 sec	0.066667	0.066667
27∼30 sec	0.066667	0.066667
30∼33 sec	0.066667	0.066667
33∼36 sec	0.066667	0.066667
36∼39 sec	0.066667	0.066667
39∼42 sec	0.066667	0.066667
Larger than 42 sec	0.066667	0.066667

Table 6.4. The CPT of Node D

- Node D (Table 6.4): The cost can be calculated in distance, energy, or time based on the task information and the current state of a robot, so it is a continuous variable. Nevertheless, we may transform a continuous variable into a discrete variable by

creating several intervals in order to simplify the difficulty of calculation. We, thus, assume there is a value for every three seconds for a total of fifteen values in this node. Because the information of a robot varies by case and with time no matter whether a robot takes the task, the conditional probability distribution is uniform.

- Node E (Table 6.5): This node represents the battery level of a robot in voltage, so it is a continuous variable from the minimum working voltage (two volts) to the maximum voltage (five volts). With the same concept in Node D, it can be changed into a discrete variable. The probability distribution is uniform because the battery level varies with time. Hence, we assume there is a value for every one volt for a total of three values in this node.

- Node F (Table 6.6): This node has two values: Yes and No. We may reasonably assume that the probability of Yes would be smaller when the cost increases or the battery level lowers. In the training process, we may collect data and update this CPT to gain more accurate conditional probabilities.

- Node J: This node has two values: Yes and No, which shows whether a robot has the skill. The probability can be calculated depending on the number of robots with and without the skill in a team. Similar to Node B, the value of the probability

E	**P(E)**
2∼3 V	0.333333
3∼4 V	0.333333
4∼5 V	0.333333

Table 6.5. The Probability Table of Node E

P(F=Yes|DE)		E		
		2∼3 V	3∼4 V	4∼5 V
D	0∼3 sec	0.900	0.950	0.999
	3∼6 sec	0.850	0.900	0.950
	6∼9 sec	0.800	0.850	0.900
	9∼12 sec	0.750	0.800	0.850
	12∼15 sec	0.700	0.750	0.800
	15∼18 sec	0.650	0.700	0.750
	18∼21 sec	0.600	0.650	0.700
	21∼24 sec	0.550	0.600	0.650
	24∼27 sec	0.500	0.550	0.600
	27∼30 sec	0.450	0.500	0.550
	30∼33 sec	0.400	0.450	0.500
	33∼36 sec	0.350	0.400	0.450
	36∼39 sec	0.300	0.350	0.400
	39∼42 sec	0.250	0.300	0.350
	Larger than 42 sec	0.200	0.250	0.300

Table 6.6. The CPT of Node F

J	**P(J)**
Yes	0.999
No	0.001

Table 6.7. The Probability Table of Node J (Driving Skill)

cannot be zero or one. Table 6.7 shows the probability of the driving skill when all robots have this skill. The probability of Yes is very close to one because all robots are ground vehicles in our experiments.

- Node K: This node has two values: Good and Bad, which represents the current

skill condition. Because it fluctuates with time, we should keep monitoring and updating the CPT. Nevertheless, the initial conditional probability may be established according to the accuracy of localization, the precision of a sensor, or the quality of a camera. On the contrary, the conditional probability of Good should be very small (close to zero) if a robot does not have the skill. Table 6.8 shows the initial conditional probability of the driving skill, and Table 6.9 represents the conditional probability of the driving skill when a robot cannot drive anymore due to accidents.

- Node G (Table 6.10): This node has two values: Yes and No. When a robot takes the task, the conditional probability of Yes could be small only when one or more than one of the skill conditions is Bad in that the robot might not have enough skills to complete the task. Contrarily, when a robot does not take the task, the probability distribution could be assumed as uniform because we are not able to determine whether the robot has enough skills. This CPT can be trained after collecting data to receive more realistic results.

J	**P(K=Good|J)**
Yes	0.800
No	0.001

Table 6.8. The CPT of Node K (Condition of Driving)

J	**P(K=Good|J)**
Yes	0.001
No	0.001

Table 6.9. The CPT of Node K (Condition of Driving) When a Robot Cannot Drive

C	K	P(G=Yes|CK)
Yes	Good	0.900
Yes	Bad	0.001
No	Good	0.500
No	Bad	0.500

Table 6.10. The CPT of Node G (One Required Skill)

C	F	G	P(H=Yes|CFG)
Yes	Yes	Yes	0.900
Yes	Yes	No	0.010
Yes	No	Yes	0.010
Yes	No	No	0.001
No	Yes	Yes	0.001
No	Yes	No	0.001
No	No	Yes	0.001
No	No	No	0.001

Table 6.11. The CPT of Node H

- Node H (Table 6.11): This node has two values: Yes and No. If a robot takes the task, the conditional probability of Yes is large only when both values of Node F and Node G are Yes. Otherwise, the conditional probability of Yes is small. This CPT can also be trained with a large amount of data to get more practical results.

6.3 The Calculation of a Success Rate

With the BN of a task and the initial CPTs, we are able to calculate the success rate of a task. According to Bayes' theorem, we can calculate the posterior probability based on the information of a BN, CPTs, and evidence (the known values of nodes). In this dissertation, the posterior probability is meaningful. The success rate of a task denotes

the posterior probability when a robot takes the task and completes it successfully. With the BN and CPTs of a task, the success rate of a task can be calculated after gathering certain evidence. In the BN, the type of robot (Node B), the current conditions of a robot (Node A and Node E), the skills of a robot (Node J and other Skill nodes), and the cost to complete the task (Node D) are known. Subsequently, in order to calculate the success rate of a task based on this evidence, we can compute the posterior probability when the values of Node C and Node H are Yes. By following Bayes' theorem, the general equation is listed in Eq. (6.1).

$$
\begin{aligned}
P(CH|ABDEJ) &= \frac{P(A,B,C,D,E,H,J)}{P(ABDEJ)} \\
&= \frac{\sum_{F,G,K} P(H|CFG)P(G|CK)P(K|J)P(F|DE)P(D|C)P(C|AB)P(A)P(B)P(E)P(J)}{\sum_{C,F,G,H,K} P(H|CFG)P(G|CK)P(K|J)P(F|DE)P(D|C)P(C|AB)P(A)P(B)P(E)P(J)} \\
&= \frac{\sum_{F,G,K} P(H|CFG)P(G|CK)P(K|J)P(F|DE)P(D|C)P(C|AB)}{\sum_{C,F,G,H,K} P(H|CFG)P(G|CK)P(K|J)P(F|DE)P(D|C)P(C|AB)}
\end{aligned}
\tag{6.1}
$$

6.4 The Application of the General Bayesian Network of a Task

After comprehending the fundamental knowledge about the general Bayesian Network of a task, the CPTs, and the calculation of a success rate, we may apply them to other tasks in a mission. When a task requires one or multiple robots with one skill, the BN of the task is the same as the general Bayesian Network (Fig. 6.1). When a task demands one or

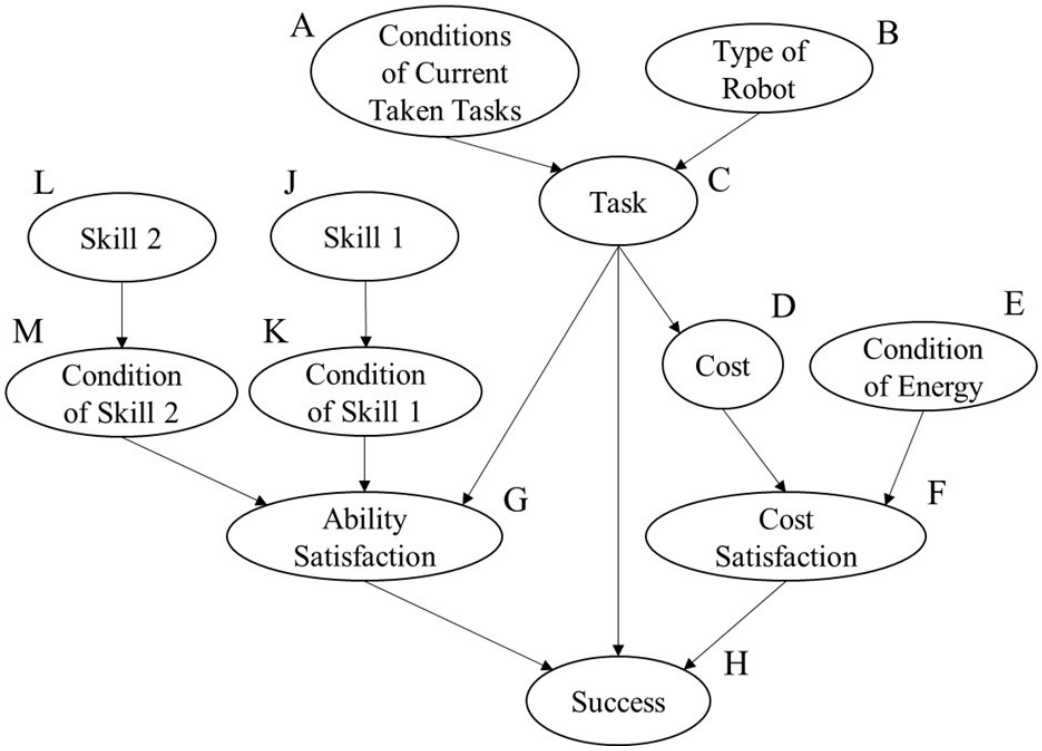

Figure 6.2. The Bayesian Network of a Task with Two Required Skills

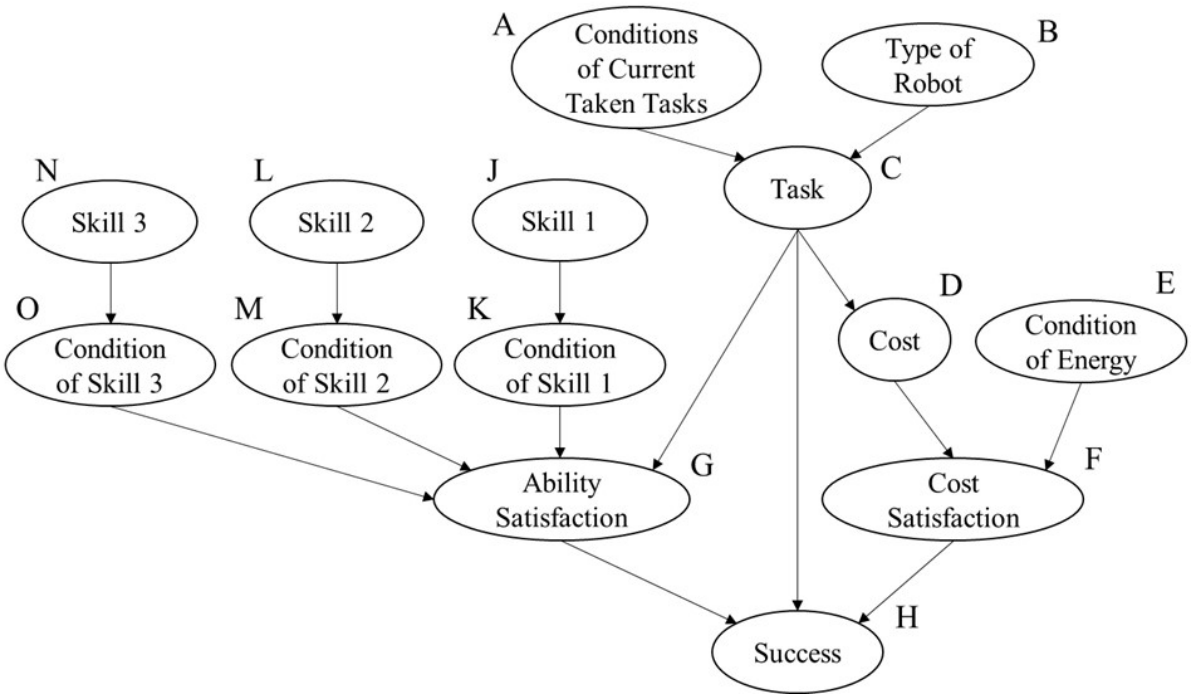

Figure 6.3. The Bayesian Network of a Task with Three Required Skills

several robots with two skills, we may increase the number of nodes related to skills (Skill and Skill Condition nodes) from the general Bayesian Network and build the BN of the task in Fig. 6.2. When a task needs one or many robots with three skills, with a similar concept, we may simply increase the number of the nodes related to skills and build the BN of the task in Fig. 6.3. As can be seen, it is very simple to create the BN of a task from the general Bayesian Network by adding a few additional nodes corresponding to the number of required skills.

After building the BNs of tasks, probabilities and initial CPTs of the BNs can be calculated and created based on the descriptions in Sec. 6.2. Nodes A, B, and E are the same in each task because they are related to robots instead of tasks. Nodes J, K, L, M, N, and O (or more nodes related to required skills) are the same in each task due to the hardware on each robot. The initial CPTs are shown in Tables 6.7, 6.8, 6.9, 6.12, 6.13, 6.15, and 6.16 where the probabilities of Node L and Node N can be calculated based on the number of robots with the skill (camera, arm, or sensor). However, the CPTs of Nodes K, M, and O (or more nodes related to skill conditions) would be different from each robot because of distinctive hardware (Tables 6.8, 6.9, 6.13, and 6.16). The CPTs of Nodes C and H could be the same in each task because they are dependent on the information of a robot. As for Node G in each task, the CPTs would be different because the number of required skills is not always the same. Nevertheless, the CPT of Node G could be identical in each task if the number of required skills is the same due to the causal relationship with skill conditions. The initial CPT of Node G with one skill is the same in Table 6.10. The initial CPT of Node G with two skills is assumed in Table

L	P(L)
Yes	0.500
No	0.500

Table 6.12. The Probability Table of Node L (Camera or Arm Skill)

L	P(M=Good|L)
Yes	0.900
No	0.001

Table 6.13. The CPT of Node M (Condition of Camera or Arm)

C	K	M	P(G=Yes|CKM)
Yes	Good	Good	0.900000
Yes	Good	Bad	0.001000
Yes	Bad	Good	0.001000
Yes	Bad	Bad	0.000100
No	Good	Good	0.500000
No	Good	Bad	0.500000
No	Bad	Good	0.500000
No	Bad	Bad	0.500000

Table 6.14. The CPT of Node G (Two Required Skills)

6.14. The initial CPT of Node G with three skills is set in Table 6.17. With respect to Node D and Node F, they are related to the difficulty of a task and the robot condition, so the CPTs of Node D and Node F in each task are individual. We, nonetheless, may create the initial CPTs based on alike concepts or with the same probabilities (Table 6.4 and Table 6.6) for each task and then individually update them after collecting data from experiments and missions.

With the BNs and CPTs of tasks, we are able to calculate the success rates of the

N	P(N)
Yes	0.999
No	0.001

Table 6.15. The Probability Table of Node N (Sensor Skill)

N	P(O=Good|N)
Yes	0.900
No	0.001

Table 6.16. The CPT of Node O (Condition of Sensor)

C	K	M	O	P(G=Yes|CKMO)
Yes	Good	Good	Good	0.900000
Yes	Good	Good	Bad	0.001000
Yes	Good	Bad	Good	0.001000
Yes	Good	Bad	Bad	0.000100
Yes	Bad	Good	Good	0.001000
Yes	Bad	Good	Bad	0.000100
Yes	Bad	Bad	Good	0.000100
Yes	Bad	Bad	Bad	0.000010
No	Good	Good	Good	0.500000
No	Good	Good	Bad	0.500000
No	Good	Bad	Good	0.500000
No	Good	Bad	Bad	0.500000
No	Bad	Good	Good	0.500000
No	Bad	Good	Bad	0.500000
No	Bad	Bad	Good	0.500000
No	Bad	Bad	Bad	0.500000

Table 6.17. The CPT of Node G (Three Required Skills)

tasks according to Bayes' theorem. The general equation (6.2) is used to calculate the success rate of a task with two required skills, and the general equation (6.3) is for the calculation of the success rate of a task with three required skills.

$$
\begin{aligned}
P(CH|ABDEJL) &= \frac{P(A,B,C,D,E,H,J,L)}{P(ABDEJL)} \\
&= \frac{\sum_{F,G,K,M} \frac{P(H|CFG)P(G|CKM)P(K|J)P(M|L)P(F|DE)}{P(D|C)P(C|AB)P(A)P(B)P(E)P(J)P(L)}}{\sum_{C,F,G,H,K,M} \frac{P(H|CFG)P(G|CKM)P(K|J)P(M|L)P(F|DE)}{P(D|C)P(C|AB)P(A)P(B)P(E)P(J)P(L)}} \\
&= \frac{\sum_{F,G,K,M} P(H|CFG)P(G|CKM)P(K|J)P(M|L)P(F|DE)P(D|C)P(C|AB)}{\sum_{C,F,G,H,K,M} P(H|CFG)P(G|CKM)P(K|J)P(M|L)P(F|DE)P(D|C)P(C|AB)}
\end{aligned}
\tag{6.2}
$$

$$
\begin{aligned}
P(CH|ABDEJLN) &= \frac{P(A,B,C,D,E,H,J,L,N)}{P(ABDEJLN)} \\
&= \frac{\sum_{F,G,K,M,O} \frac{P(H|CFG)P(G|CKMO)P(K|J)P(M|L)P(O|N)P(F|DE)}{P(D|C)P(C|AB)P(A)P(B)P(E)P(J)P(L)P(N)}}{\sum_{C,F,G,H,K,M,O} \frac{P(H|CFG)P(G|CKMO)P(K|J)P(M|L)P(O|N)P(F|DE)}{P(D|C)P(C|AB)P(A)P(B)P(E)P(J)P(L)P(N)}} \\
&= \frac{\sum_{F,G,K,M,O} P(H|CFG)P(G|CKMO)P(K|J)P(M|L)P(O|N)P(F|DE)P(D|C)P(C|AB)}{\sum_{C,F,G,H,K,M,O} P(H|CFG)P(G|CKMO)P(K|J)P(M|L)P(O|N)P(F|DE)P(D|C)P(C|AB)}
\end{aligned}
\tag{6.3}
$$

By following the same steps, we can build BNs and CPTs and calculate the success rates of tasks with more than three required skills although they are not described in this dissertation.

6.5 The Training Process of Bayesian Networks (BNs)

As we know, a BN is a machine-learning approach, so the BNs of tasks can be trained after collecting a vast amount of data from real experiments or missions. Learning the

structure of a BN is negligible in this dissertation since the general Bayesian Network of a task and the BNs of tasks are built based on obvious, meaningful causal relationships between factors (nodes). We should not easily modify the structures of the BNs according to the collected data. On the contrary, even though initial CPTs are reasonably designed, the success rate of a task would be unrealistic. In order to receive near-optimal, more accurate results, updating CPTs in a BN, hence, is crucial in the training process.

The first step of the training process is to collect data from hardware experiments. The values of Nodes A, B, C, D, E, H, J, L, and N (and other Skill nodes) can be known as we may assign one task to a robot and the robot may or may not complete the task in experiments. The values of Nodes F, G, K, M, and O (and other Skill Condition nodes) could not be determined when a robot executes the task because skill conditions keep changing with time and there is no algorithm to check if required skills or remaining energy is satisfied. As a result, collected data would be incomplete data (missing some values).

The second step is to update the CPTs in a BN with collected data. The most common, efficient approach to learning from a large data set is the maximum likelihood (ML) approach or the maximum likelihood (ML) parameter estimate. However, in our situation, the collected data is incomplete. To learn from the incomplete data, we need to utilize expectation maximization (EM) first to calculate the conditional probability of the missing component in each case and then apply the ML approach [156].

6.5.1 Expectation Maximization (EM)

The general concept of EM is to obtain the expected empirical distribution of the incomplete data (**D**) based on the initial CPTs in a BN. With EM, we are able to calculate the conditional probability of the missing component with original CPTs in each case. The general equation is listed in Eq. (6.4). With this equation, the conditional probability of the missing component in each case can be found and then can be used in the ML approach. Examples are shown in the case study in Chapter 10.

$$Pr(c_i|\mathbf{d_i}) = \frac{Pr(c_i, \mathbf{d_i})}{Pr(\mathbf{d_i})} \tag{6.4}$$

where $\mathbf{d_i}$ is a case in the incomplete data (**D**) and c_i means the factors with missing values in the case

6.5.2 The Maximum Likelihood (ML) Approach

The broad notion of the ML approach is to estimate and update the CPTs of a BN according to collected data. The general equation to update CPTs is in Eq. (6.5). With this equation, each updated conditional probability in a CPT can be found with the collected data set. In addition, to avoid the problem that the updated conditional probability is zero or one, a certain value that is very close to zero or one could be defined as the updated conditional probability.

$$P^{k+1}(x|u) \equiv Pr_{\mathbf{D},P^k}(x|u) = \frac{Pr_{\mathbf{D},P^k}(x,u)}{Pr_{\mathbf{D},P^k}(u)} = \frac{\sum_{i=1}^{N} Pr_{P^k}(x,u|\mathbf{d_i})}{\sum_{i=1}^{N} Pr_{P^k}(u|\mathbf{d_i})} \tag{6.5}$$

where x and u are the values of the factors in a BN, P^k means the original conditional probabilities in CPTs, P^{k+1} means the updated conditional probability in a CPT, and N is the number of the cases in the incomplete data ($\mathbf{D}$).

After receiving the updated CPTs with Eq. (6.5), the learning process is not done yet. According to the theory of the maximum likelihood, the updated likelihood is always greater than or equal to the original likelihood in Eq. (6.6), which is also true for the log-likelihood in Eq. (6.7). As a result, the CPTs must be updated through many iterations with the same collected data until the result converges, so we may set up a threshold (a value) to stop the training process. Examples are shown in the case study in Chapter 10.

$$L_{k+1}(\mathbf{D}) = \prod_{i=1}^{N} Pr_{k+1}(\mathbf{d_i}) \geq L_k(\mathbf{D}) = \prod_{i=1}^{N} Pr_k(\mathbf{d_i}) \tag{6.6}$$

where L_k is the original likelihood, L_{k+1} is the updated likelihood, and N is the number of the cases in the data ($\mathbf{D}$).

$$LL_{k+1}(\mathbf{D}) = \sum_{i=1}^{N} \log\left[Pr_{k+1}(\mathbf{d_i})\right] \geq LL_k(\mathbf{D}) = \sum_{i=1}^{N} \log\left[Pr_k(\mathbf{d_i})\right] \tag{6.7}$$

where LL_k is the original log-likelihood, LL_{k+1} is the updated log-likelihood, and N is the number of the cases in the data ($\mathbf{D}$).

6.6 The Optimal Combination of MRTA

After calculating the success rates of tasks no matter whether using the initial CPTs or the trained CPTs, we are able to allocate the tasks based on the values of the success rates. In order to obtain the near-optimal solution of MRTA, the proposed method

in this dissertation is a hybrid approach combining an auction-based approach and an optimization-based approach. According to the literature review in Chapter 3, there are pros and cons to optimization-based approaches and auction-based approaches, whereas a better result could be achieved by combining them. It is simple to receive the optimal solution after collecting data from robots with an optimization-based approach although a single-point failure may happen in a centralized MRS. On the other hand, an auctioneer is not fixed in a decentralized or distributed MRS with an auction-based approach while the result of MRTA might be sub-optimal. Accordingly, an auction-based approach is used to collect success rates from robots, and an optimization-based approach is utilized to obtain the near-optimal solution based on the gathered success rates no matter whether in a centralized, decentralized, or distributed MRS in our approach.

In a centralized MRS, an auctioneer is fixed on the central controller (or one robot) assigned by a designer. In a decentralized or distributed MRS, an auctioneer is dynamically assigned to one of the robots in a team. During the Announcement Stage, the auctioneer announces all tasks in a mission to all robots. Then, each robot calculates the success rate of each task and sends all the bids (the success rates) to the auctioneer in the Submission Stage. After the auctioneer collects the success rates from all robots, in the Selection Stage, an optimizer (like GA or AC) is used to gain the near-optimal result according to the objective function and rules (constraints). The objective function is in Eq. (6.8). The goal is to find the maximum sum of the average success rate of each task in order to benefit most robots and the whole team. Additionally, five rules (Rules I to V) should be followed during the process.

$$P^* = \arg\max_{P \in P_{ij}} \left(\sum_{j=1}^{n} \bar{P}_j \right), \ \bar{P}_j = \frac{\sum P_{ij}}{m} \tag{6.8}$$

where P_{ij} is the success rate (the posterior probability) when the robot i takes the task j, m is the number of chosen robots in the task j, and n is the number of tasks.

I. One ST or MT robot can only choose one task or none of them in a mission.

II. Depending on the different types of tasks (SR or MR), the number of robots that choose the same task is limited.

III. Each chosen success rate (posterior probability) must be larger than a certain value (a threshold) to consider as a robot may successfully complete a task.

IV. Without the violation of Rule III, all tasks must be allocated as long as the total number of robots is larger than or equal to the total number of robots the tasks require.

V. Without the violation of Rule III, all robots must be assigned as long as the total number of robots the tasks require is less than or equal to the total number of robots.

In order to implement our approach in most applications, one more rule (Rule VI) is added in this process when considering the priorities of tasks.

VI. Tasks with higher priorities must be allocated without the violation of Rule III.

As can be seen, only one objective function related to success rates is considered in this approach for the reason that dynamic factors have already been included in the BNs of tasks, such as robot conditions, skill conditions, and costs to complete the tasks.

6.7 The Comparison of MRTA Approaches

After comprehending the novel approach with BNs, we may compare our technique and current MRTA methodology with many aspects. Different approaches have advantages and disadvantages and may be utilized in different applications, so it is difficult to distinguish which strategy is better. Thus, we organize the characteristics of each approach in Table 6.18 according to research papers [90, 95, 97, 106] and allow readers to choose the one which is preferred in their applications.

According to the information in Chapter 3, behavior-based approaches are usually used in a heterogeneous, distributed MRS, whereas structures are hard to design; optimization-

	L-ALLIANCE [95] (behavior-based)	**PSO [97] (optimization-based)**	**Trader Bots [106] (auction-based)**	**Proposed Approach**
MRS	distributed	centralized	decentralized/ distributed	centralized/ decentralized/ distributed
Heterogeneity	Good	Bad	Normal	Good
Environment	dynamic	static	dynamic	dynamic
Computation	O(mn)	O(mn)/controller	O(1)/bidders O(n)/auctioneer	O(m)/bidders O(mn)/auctioneer
Communication	O(m)	O(n)/controller O(1)/robots	O(n)	centralized: O(n)/controller O(1)/robots decentralized: O(n)/coordinators O(1)/robots distributed: O(n)
Formalization	No	Yes	No	Yes
Solution Quality	sub-optimal	optimal	sub-optimal	near-optimal
Scalability	Bad	Good	Normal	Normal $\sim$ Good
Learning	Yes	No	No	Yes
n is the number of robots, and m is the number of tasks. The complexity of computation and communication is calculated per iteration.				

Table 6.18. The Comparison of the MRTA Approaches

based approaches are ordinarily utilized in a homogeneous, centralized MRS while cost functions vary by cases; auction-based approaches are typically exploited in a decentralized or distributed MRS although costs and rewards are customized. On the contrary, our approach could be implemented in three types of MRSs. Skills (capabilities) and dynamic conditions in BNs and the success rates of tasks are considered, so our approach could be applied to most MRSs. With respect to the difficulty of computation, the complexity of communication, and the solution quality, our method is the combination of an auction-based approach and an optimization-based approach, so the result would be similar to theirs. The complexity of communication would be the same as optimization-based approaches in a centralized MRS or identical to auction-based approaches in a distributed MRS. In a decentralized MRS, the complexity of communication is in between depending on the number of coordination robots. Thus, the complexity of communication, basically, is O(n) in our approach. The difficulty of computation for bidders is O(m) because each robot should calculate the success rate of each task. The difficulty of computation for an auctioneer is O(mn) now that the auctioneer utilizes an optimizer to obtain the near-optimal solution. Although our approach seems to be more complicated, the result is better. Unlike the structure design in behavior-based approaches and the cost and reward in auction-based approaches, BNs, CPTs, and the objective function are systematically established instead of trial and error. Moreover, the solution is near-optimal rather than sub-optimal. Only one iteration is required in the MRTA process like an optimization-based approach instead of several iterations in an auction-based approach. Different from optimization-based approaches, our method can be applied to a heterogeneous robot team

under dynamic environments, and only one objective function including certain rules (constraints) is involved. Regarding scalability, we may estimate that the scalability of our method is between optimization-based approaches and auction-based approaches because of a hybrid approach. Last but not least, our approach is learnable. A better or optimal result might be obtained after collecting data from experience.

Chapter 7

The Generic Agent-based Framework in Multi-Robot Systems (MRSs)

In this chapter, the structure of the generic agent-based framework of MRSs is going to be introduced. Our goal is to design a framework that can be broadly applied to different types of MRSs and applications. Researchers are able to utilize various algorithms to control hardware, exchange and fuse information, sense surroundings, coordinate robots, and make adequate decisions. Consequently, we state crucial commons from diverse MRSs in numerous applications and design agents divided with agencies in the generic agent-based framework.

7.1 Database (DB) or Knowledge Base (KB)

A database (DB) or knowledge base (KB) is used to store data or information for agents to access. Even though agents may directly exchange data or information among them, they may also read, store, and exchange data or information through a DB or KB, such as map information, task information, robot information, planning information, rough data received from robots or hardware, and other information related to decision making.

CPTs are also saved in the DB or KB in our approach. The type of file to store data or information can be TXT, DAT, CSV, XML, JPEG, MP3, or MP4 depending on the design.

7.2 Decision-making Agency

This is a huge agency where agents are utilized to coordinate robots, allocate tasks, and control robot motions. Basically, we may divide these agents into two groups based on the main purpose: coordination and planning.

The first group of agents, coordination agents, is responsible for high-level decision-making. Robots may allocate tasks and coordinate robots to complete received tasks individually or cooperatively. In order to illustrate our MRTA approach in this agent-based framework, the task allocation agent is intentionally specified although it is one of the coordination agents. According to current MRTA techniques, one agent, the task allocation agent, executes one program to deal with task allocation. In optimization-based approaches, the agent runs one optimizer (like GA or AC) to allocate tasks to robots after collecting essential data. In auction-based approaches, one auctioneer (one agent) selects winners after receiving bids from robots and assigns tasks to them. Nevertheless, one task allocation agent might not be enough in complicated methods. For instance, one auctioneer, in auction-based approaches, must be selected at the beginning of task allocation, and costs or rewards in algorithms need to be calculated before or during the MRTA process. Consequently, task allocation assistant agents are designed. In our approach, these agents are used to calculate success rates and to update CPTs after collecting data from hardware (skills) besides selecting an auctioneer or resuming an

MRTA process. They can also be utilized to update CPTs during an onboard learning process. Therefore, assistant agents including task allocation assistant agents are required to help coordination agents including task allocation agents calculate and store parameters used in sophisticated algorithms. As a result, coordination agents and assistant agents are essential in the decision-making agency although a state-of-the-art coordination method is not proposed in this dissertation except for the novel MRTA approach.

The other group of agents, planning agents, is responsible for low-level decision-making. After a robot receives a task, a planning agent should start planning how to complete this task individually or cooperatively. The robot may use deliberative methods, such as A-star (A*), Artificial Potential Field (APF), rapidly exploring random trees (RRT), GA, PSO, and AC, or reactive methods to avoid obstacles [73, 83, 160]. Moreover, it may avoid collisions with other robots or dynamic barriers with fuzzy logic [161] or machine-learning methods to cope with uncertainty. With a similar concept of coordination agents, other assistant agents, accordingly, are needed to help planning agents for motion planning when complex algorithms are applied like machine-learning methodology. A modern motion-planning approach, however, is not implemented in this dissertation. To sum up, coordination agents, planning agents, and assistant agents are crucial in the decision-making agency.

7.3 Information Agency

In this agency, agents are utilized to process information. No matter what type of MRS is or what abilities robots have, information must be analyzed, exchanged between each robot, fused, and saved in the DB or KB. All agents in this agency can be named infor-

mation processing agents. To provide more details, perception agents, information fusion agents, and information exchange agents are classified from information processing agents. Robots with sensors or cameras may perceive environments and save data or images in the DB or KB. Then, the perception agents are used to analyze certain features from the data and detect objects with machine-learning approaches or Bayesian Inference [162]. In addition, robots need to exchange their information to recognize surroundings, allocate tasks, coordinate robots, and generate proper behaviors or reactions, so the information exchange agents are utilized to send and receive important information among robots. Numerous examples are robot lists, task lists, assigned tasks, maps, the poses or locations of robots, and success rates in our approach. After collecting information from other robots and the DB or KB, the information fusion agents may fuse former and present or internal and external information to accurately estimate current states and to obtain up-to-date data with various approaches, such as Kalman Filter [163], Split Covariance Intersection Filter [164, 165], and machine-learning methods. Due to advanced methodology, assistant agents are added because they might be required to help perception agents and information fusion agents to calculate or update parameters. Accordingly, all the information processing agents can execute a diversity of programs depending on researchers' requirements and techniques, which means various approaches can be applied to process received information, save significant results in the DB or KB, and exchange information between agents.

7.4 Execution Agency

In this agency, agents are utilized to control and inspect hardware or devices. Every robot may have diverse hardware equipped, such as motors with wheels or propellers, sensors, cameras, or arms, so a number of agents in this agency are included to control them. Each agent should handle each device with a low-level controller. For instance, one agent executes one program to control the wheels on a robot to drive the robot to the specified position after receiving a command from a high-level controller. Another agent runs another program to take pictures with a camera when necessary. Therefore, several agents, which are called hardware control agents, handling distinctive programs are used to control numerous devices in the Execution Agency. Furthermore, some devices may deteriorate with time in dynamic environments, so condition detection agents are needed to run other programs or algorithms to detect or determine whether the equipment works well like Kullback-Leibler Divergence [166] apart from machine-learning approaches. With the same viewpoint, assistant agents might be required to help condition detection agents with sophisticated approaches. During the executions of programs, all agents may save data or information in the DB or KB for other agents' use or directly send required data to other agents.

7.5 The Generic Framework Structure

The overall structure of the proposed generic agent-based framework is shown in Fig. 7.1. Backup agents are additionally added in each agency to increase the robustness of the framework to take over essential jobs when crucial agents fail temporarily [82]. Based on this generic agent-based framework, we are able to build centralized, decentralized,

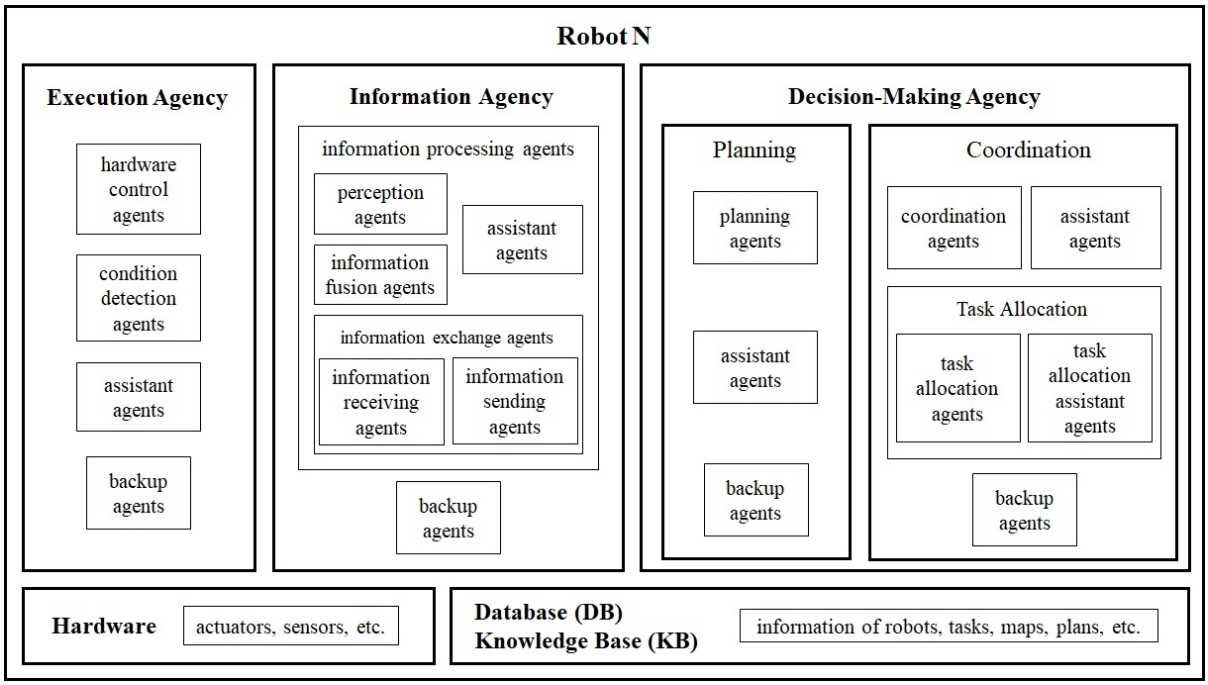

Figure 7.1. The Generic Agent-based Framework

and distributed MRSs with various approaches or algorithms to deal with distinctive situations in diverse applications. For example, a central controller in a centralized MRS or coordinators in a decentralized MRS may require coordinate agents to make high-level decisions. Coordination agents for other robots may be trivial. On the other hand, each robot in a distributed MRS requires coordination agents because each robot should have the ability to make decisions and complete tasks with or without communication. More information is going to be depicted in the next section. Lastly, dataflow is flexible in the generic agent-based framework now that data or information can be freely transferred between agents or stored in the DB or KB.

7.6 The Application of the Generic Agent-based Framework

In this section, the examples of centralized, decentralized, and distributed MRSs implemented in our experiments and case studies are depicted based on the proposed generic agent-based framework. These structures are merely examples, not unique frameworks. Agents, first, are introduced in each agency, and then frameworks are shown for centralized, decentralized, and distributed MRSs.

Execution Agency

- run_camera agent: It is used to control a camera on a robot, take a picture, and save the photo as a JPG file in the DB.
- run_linkbot agent: It is used to control the motion of a hardware robot and store current robot states in the DB with TXT files.
- run_sensor agent: It is used to control a sensor (an IR range sensor) on a robot and save a TXT file with measurement data in the DB.
- driving_condition_detection agent: It is used to check the condition of driving and store the result in the DB with a TXT file.

Information Agency

- map_update agent: It is used to create and save an updated map as a TXT file in the DB when the current map is different from the stored map.
- target_detection agent: It is used to detect the target from a picture and store the result in a TXT file in the DB.

- map_fusion agent: It is used to fuse the map when receiving different maps and update the original map in the DB.

- task_list_fusion agent: It is used to fuse the task list when receiving different task lists and update the original task list in the DB.

- receive_coop_robot_list agent: It is used to receive a robot list where robots take the same task and save the list as a TXT file in the DB.

- receive_execute_task agent: It is used to receive a signal to execute the current task and send the signal to the planning agent.

- receive_map agent: It is used to receive a map and save the received map in the DB with a TXT file.

- receive_need_task_allocation agent: It is used to receive a signal to begin the MRTA process and send the signal to the task allocation agent.

- receive_task agent: It is used to receive a task and save the task in a TXT file in the DB.

- receive_task_list agent: It is used to receive a task list and store the received task list as a TXT file in the DB.

- send_auctioneer_selection agent: It is used to send a signal to join the auctioneer-selection process.

- send_pose agent: It is used to send the current pose to other robots.

- send_success_rate agent: It is used to send success rates to other robots.

Decision-making Agency

- planning agent: It is used to generate the proper motion of a robot based on current information.
- coordination agent: It is used to coordinate robots to execute a cooperative task.
- task_allocation agent: It is used to collect success rates and allocate tasks to robots based on our approach.
- auctioneer_selection agent: It is used to select the auctioneer at the beginning of the MRTA process.
- CPT_driving_update agent: It is used to update and save the CPT of the driving skill as a TXT file in the DB.
- re_task_allocation agent: It is used to determine whether the MRTA process should start according to current conditions; for example, the success rate of the current, taken task is lower than the threshold.
- success_rate_calculation agent: It is used to calculate the success rates of tasks and store the result in the DB with TXT files.

With the agents' names and their purposes, frameworks can be drawn according to the generic agent-based framework. The centralized agent-based framework is shown in Fig. 7.2 and Fig. 7.3. Compared with these two figures, the framework installed on the central controller includes agents related to coordination, whereas the framework on other

robots excludes these agents. The decentralized agent-based framework is shown in Fig. 7.4 and Fig. 7.5. With a similar notion, the framework for coordinators (team leaders) contains agents related to coordination while the framework for other robots does not. The distributed agent-based framework is shown in Fig. 7.6, Fig. 7.7, and Fig. 7.8. The framework is the same for robots with the same skills. Certain agents related to hardware and maps are added depending on the requirements of different applications. As can be seen, numerous agent-based frameworks can be easily built from the generic agent-based framework for various applications based on diverse situations. Notwithstanding, it is emphasized that these agent-based frameworks are not singular but mere examples.

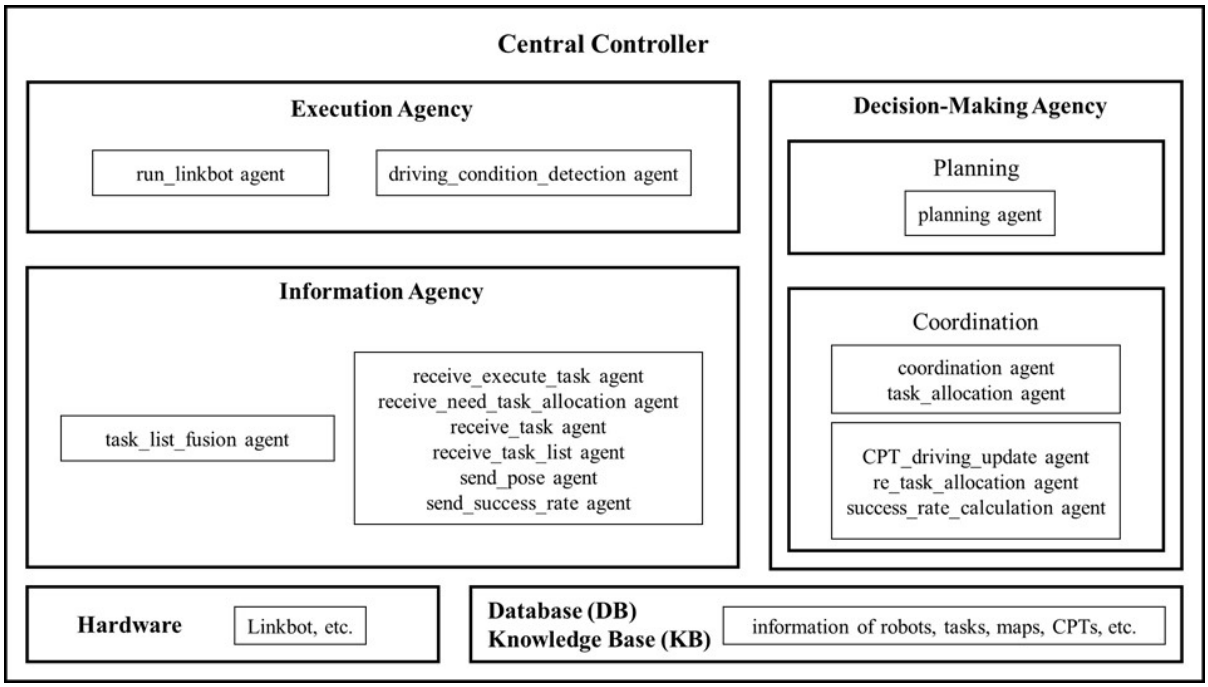

Figure 7.2. The Centralized Agent-based Framework (the Central Controller)

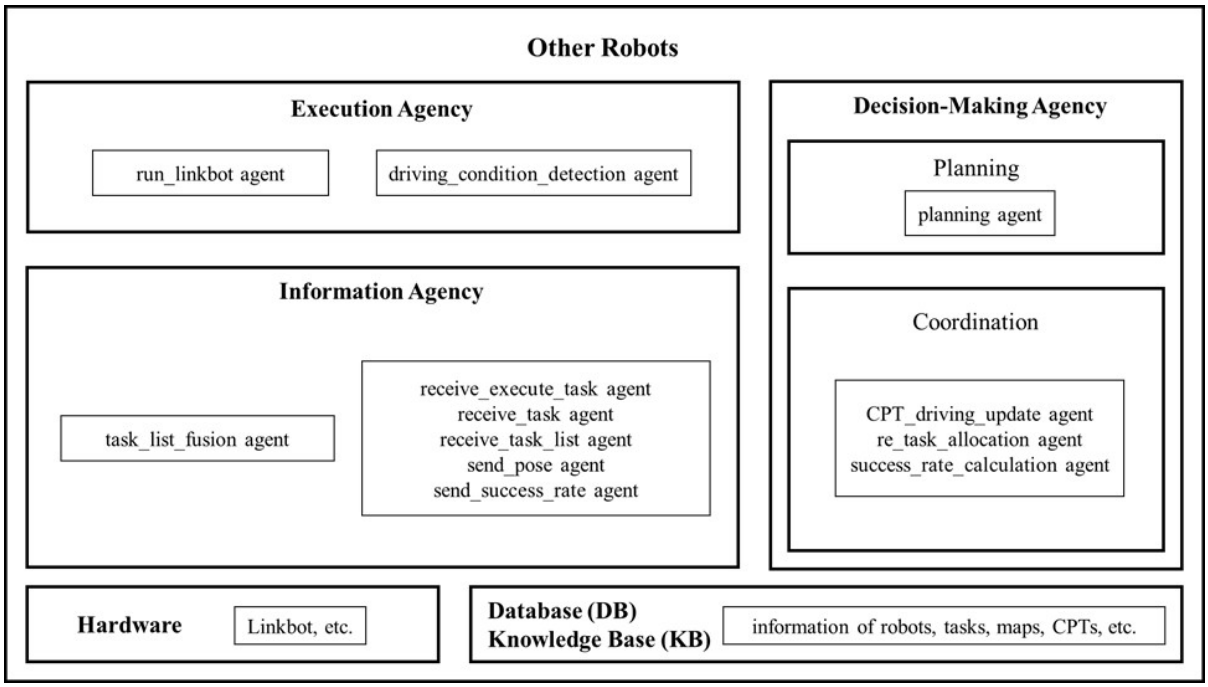

Figure 7.3. The Centralized Agent-based Framework (Other Robots)

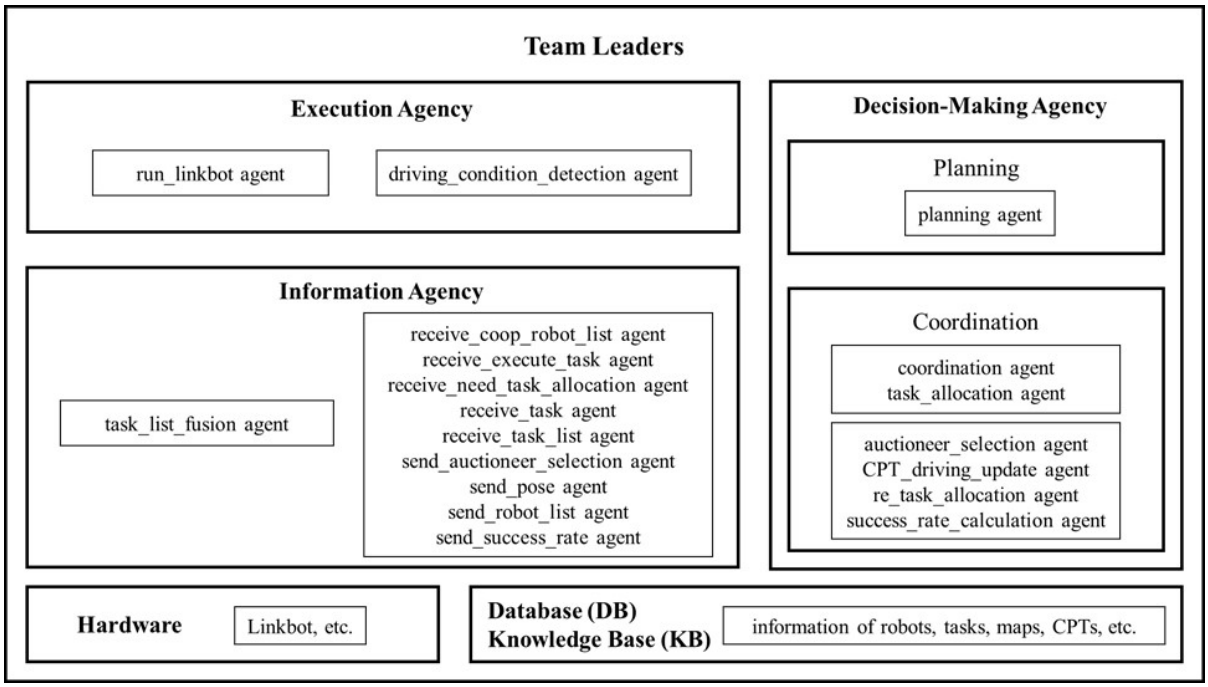

Figure 7.4. The Decentralized Agent-based Framework (Coordinators)

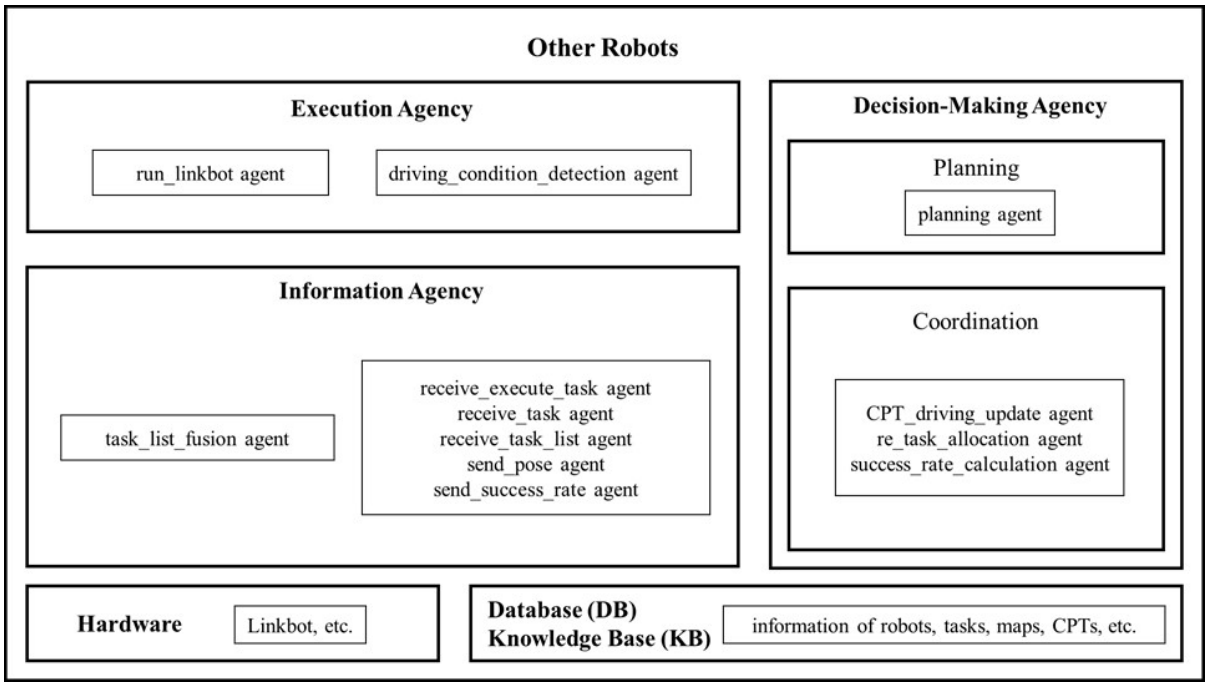

Figure 7.5. The Decentralized Agent-based Framework (Other Robots)

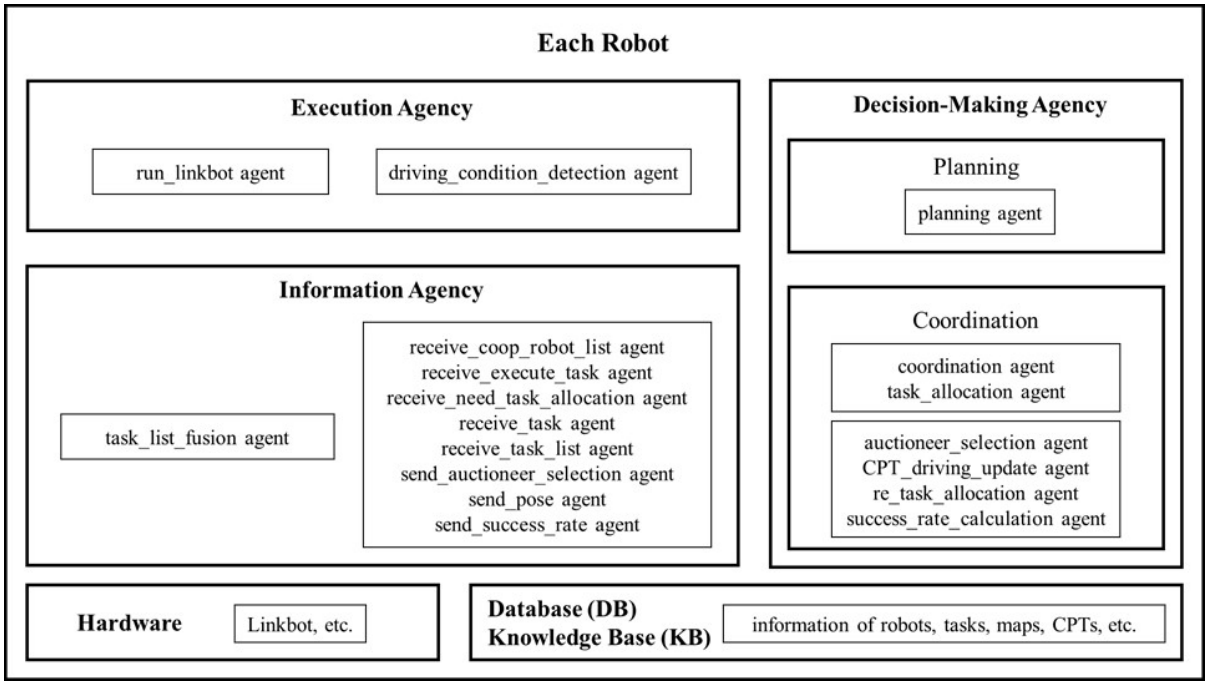

Figure 7.6. The Distributed Agent-based Framework (Comprehensive Experiments)

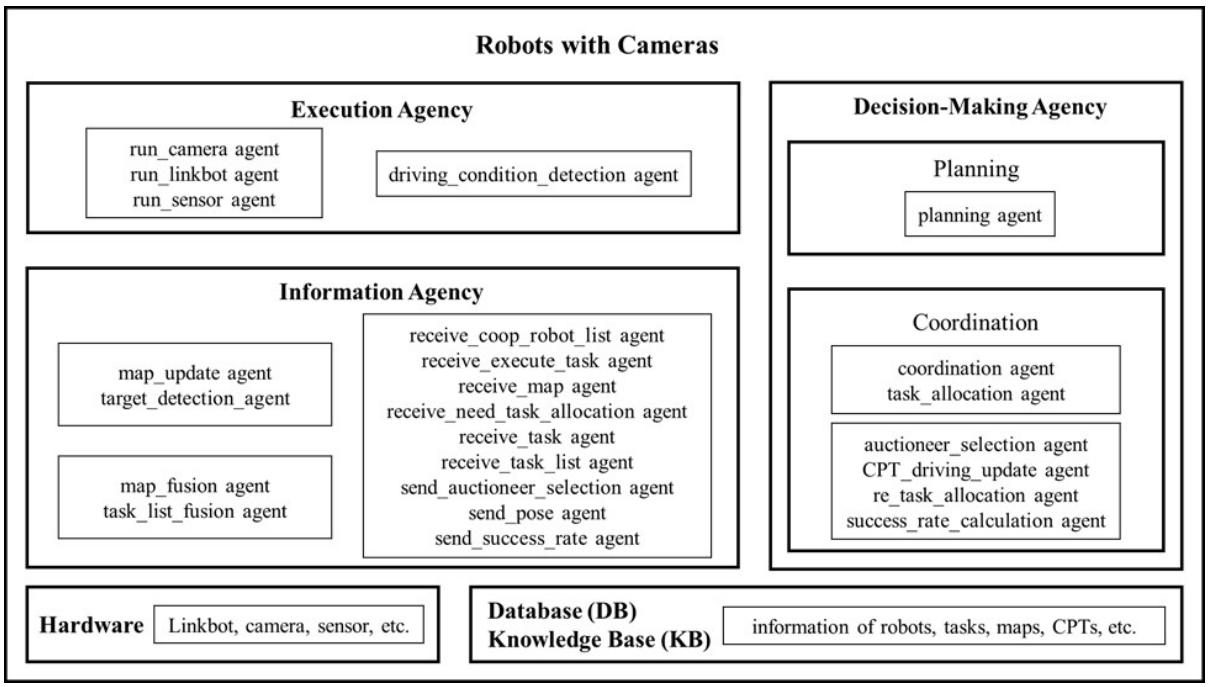

Figure 7.7. The Distributed Agent-based Framework (Robots with Cameras in Case Study)

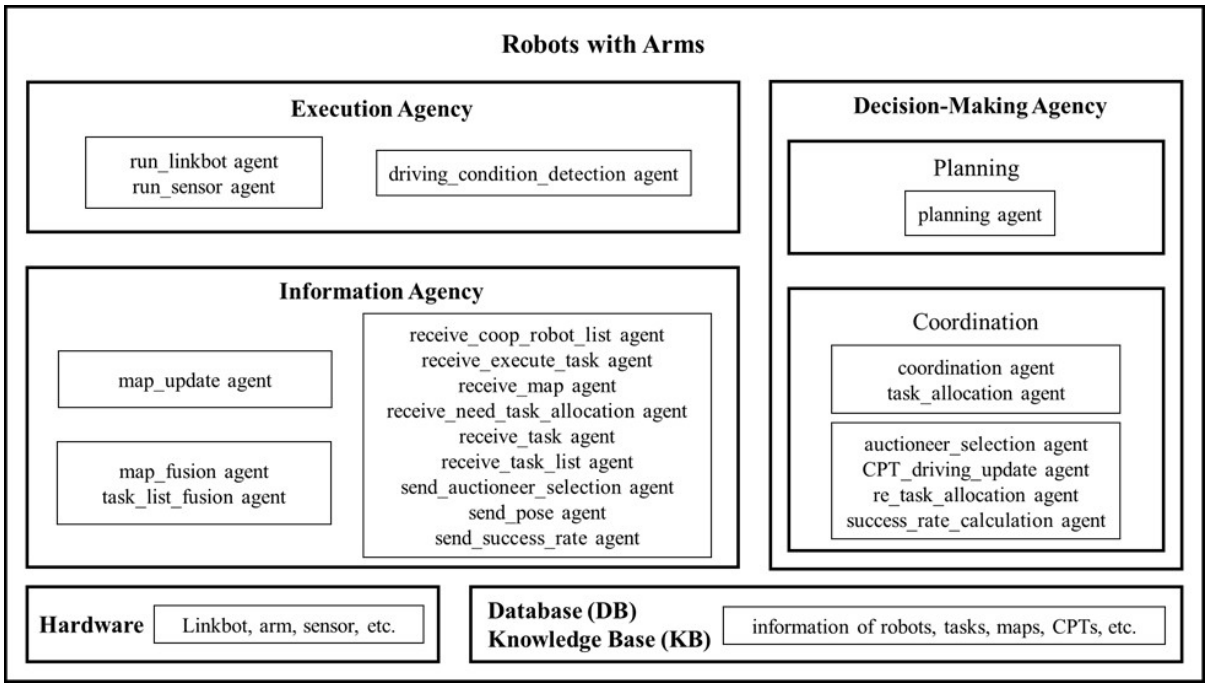

Figure 7.8. The Distributed Agent-based Framework (Robots with Arms in Case Study)

7.7 The Comparison of MRS Structures

After comprehending the generic agent-based framework, we may compare the current structures of MRSs with our framework. A group of researchers proposed a centralized MRS in [81]. They designed hardware server modules, control modules, executive modules, and interface modules in the system. Hardware server modules were utilized to control actuators and sensors, which means hardware control agents in our framework work identically. Control modules were responsible for information fusion and motion planning, so information fusion agents and planning agents may do the same jobs. Executive modules were used to make high-level decisions, which is similar to coordination agents including task allocation agents in our framework. Interface modules were graphical user interfaces (GUIs) to interact robots with users. Although we are not able to replace this module with our framework because not all software frameworks have a human-machine interface like Mobile-C, we may utilize information exchange agents to collect data on a central computer to mimic interface modules. Thus, a compatible framework may be built with our framework.

A decentralized MRS was presented in [167]. Although a complete framework was not provided, crucial features were introduced. An approach, decentralized data fusion (DDF), was proposed to fuse information from sensors; accordingly, we may use hardware control agents, information fusion agents, and information exchange agents to meet this purpose. An MRTA approach, auctioned POMDP, was designed to solve role-based MRTA problems. Information exchange agents, coordination agents, and planning agents including assistant agents may be used to choose an appropriate role and schedule proper

motions. Thereafter, we may control a robot with hardware control agents to actuate as we required. Hence, an agent-based framework for this MRS is created based on our framework.

In [80], it is a distributed MRS. Three layers and one middleware were presented in this system: hardware, intra-robot processing, distributed processing, and ARCADE middleware. The hardware layer is used to control actuators and sensors, which means hardware control agents may replace it. Because the intra-robot processing layer is responsible for task execution and information fusion, using information fusion agents and planning agents may work as well. The distributed processing layer is developed for high-level decision-making, so utilizing coordination agents including task allocation agents may achieve the same goal. The ARCADE middleware is in charge of communication, which is identical to information exchange agents. In consequence, with our generic agent-based framework, we are able to rebuild this distributed MRS.

Based on the current structure information, the implementation of our novel MRTA approach is almost impracticable in that no modules, agents, or layers were designed to check hardware conditions and update parameters for decision-making. Nonetheless, with the generic agent-based framework, we may efficiently apply the innovative MRTA algorithm and construct architectures to control hardware robots in diverse MRSs. As can be seen, with the proposed generic agent-based framework, centralized, decentralized, and distributed MRSs in most research papers could be regenerated, and adequate MRS structures for cutting-edge methodology could be devised.

Chapter 8

Comprehensive Experiments

In this chapter, we demonstrate our novel MRTA approach and the generic agent-based framework under various situations after learning the in-depth method to solve MRTA problems and build diverse MRSs. A task list and a robot list are designed to validate the proposed MRTA approach, and centralized, decentralized, and distributed MRSs are constructed from the proposed generic agent-based framework. Four essential experiments are done with hardware, low-cost robots under dynamic circumstances in three types of MRSs [168]. The videos of the comprehensive experiments can be found in [169].

8.1 Problem Statements

In order to verify whether our approach could handle different types of MRTA problems (ST/MT-SR/MR-IA) under dynamic conditions and establish numerous MRSs (centralized, decentralized, and distributed), a fabricated task list is generated in Table 8.1, and robots with different skills are designed in Table 8.2 to simulate a simple search-and-rescue mission. According to the information about the tasks (Table 8.1), Task 1, Task 2, Task 3, and Task 6 are SR tasks, and Task 4 and Task 5 are MR tasks requiring two robots

	Task Type	Required Robots	Required Skills	Target Point	Priority	Released Time
Task 1	SR	1	Driving, Camera	(20, 20)	High	First
Task 2	SR	1	Driving, Camera	(0, -20)	Medium	Second
Task 3	SR	1	Driving, Arm	(20, 0)	High	First
Task 4	MR	2	Driving, Arm	(-20, 0)	Medium	Second
Task 5	MR	2	Driving	(20, -20)	Low	Last
Task 6	SR	1	Driving	(0, 20)	Low	Last

Table 8.1. The Task List of the Comprehensive Experiments

	Robot Type	Skills	Battery Level	Initial Position	Driving Speed
Robot 1	ST/MT	Driving, Camera	3.3 V	(0, -10)	2 in/s
Robot 2	ST	Driving, Camera	3.1 V	(10, -10)	2 in/s
Robot 3	ST	Driving, Camera	3.2 V	(10, 10)	2 in/s
Robot 4	ST/MT	Driving, Arm	3.3 V	(-10, -10)	2 in/s
Robot 5	ST	Driving, Arm	3.1 V	(-10, 10)	2 in/s
Robot 6	ST	Driving, Arm	3.2 V	(0, 10)	2 in/s

Table 8.2. The Robot List of the Comprehensive Experiments

per task. Thus, this MRTA problem is an SR/MR problem. Task 1 and Task 3 with high priority are released at the beginning, Task 2 and Task 4 with medium priority are announced later, and Task 5 and Task 6 with low priority are released at the end. Hence, the task list is changeable. According to the robot list (Table 8.2), robots have different skills, so this is a heterogeneous robot team. Furthermore, the map is a known open area but dynamic, which means mobile robots may fail due to any reasons like malfunctioning hardware or obstacles. Last, three types of MRSs should be utilized to solve this MRTA problem to show how our approach could be applied to several applications.

8.2 Experiment Setup

8.2.1 Hardware and Software

Each low-cost hardware robot is composed of a Raspberry Pi and a Linkbot. A Linkbot [1] is an educational robot that can drive to a target point. A Raspberry Pi [2] is a single-board computer controlling each robot and communicating between robots. Regarding software, Mobile-C [3] is used to build agent-based frameworks for three types of MRSs. These robots may send and receive messages through the Internet via Mobile-C. Programs are written in Ch, which is a C/C++ interpreter [4]. An A* algorithm is utilized to find the shortest path to a target point.

8.2.2 The Bayesian Networks of the Tasks

With the task list of the mission, the BNs of the tasks can be built based on our approach. Task 1 requires two skills (driving and camera), so the BN of Task 1 can be created as Fig. 8.1. The BN of Task 2 is very similar to the BN of Task 1 except for the task name. The BN of Task 3 which needs two skills (driving and arm) can be generated as Fig. 8.2. The BN of Task 4 is almost the same as the BN of Task 3 except for the name of the task. Task 5 requires only one skill (driving), and its BN is shown in Fig. 8.3. With the same notion, the BN of Task 6 can be created as Fig. 8.3 except for the task name. The initial CPTs of the nodes in these BNs are identical to the CPTs described in Chapter 6 excluding the probability of Node B because the number of ST and MT robots is different in centralized, decentralized, and distributed MRSs. Details will be given later. The cost

[1] Barobo Official Website: https://www.barobo.com/
[2] Raspberry Pi Official Website: https://www.raspberrypi.com/
[3] Mobile-C Official Website: http://www.mobilec.org/
[4] Softintegration Official Website: https://www.softintegration.com/

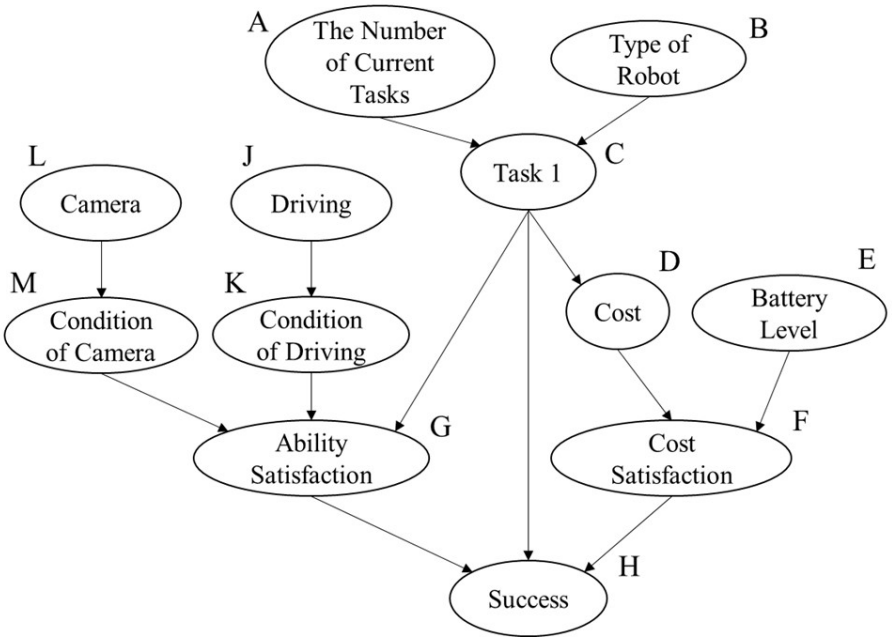

Figure 8.1. The BN of Task 1 in the Comprehensive Experiments

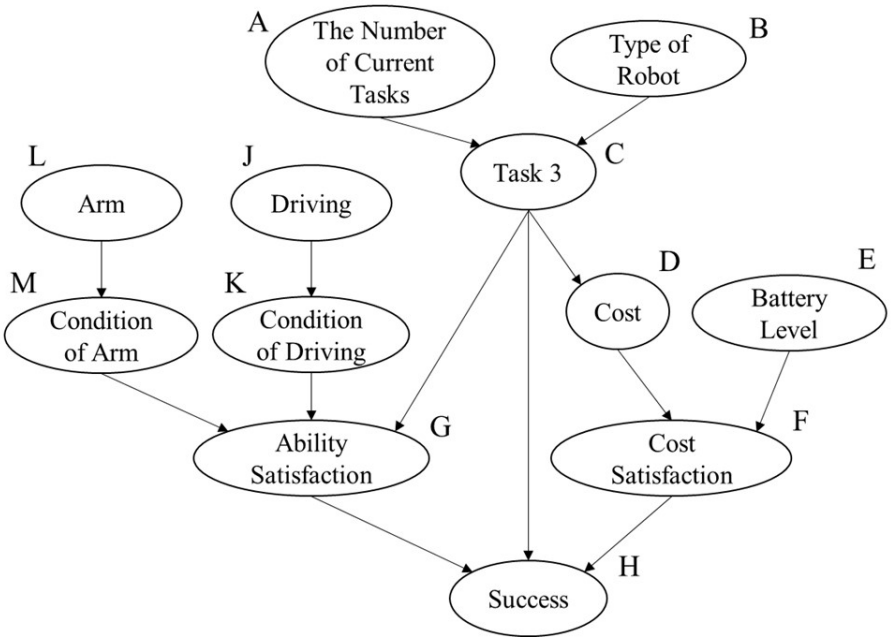

Figure 8.2. The BN of Task 3 in the Comprehensive Experiments

of a task is calculated in the seconds a robot spends from the current position to the target point. Lastly, the value of the threshold of the success rate is 0.05 to assume a robot might successfully complete a task.

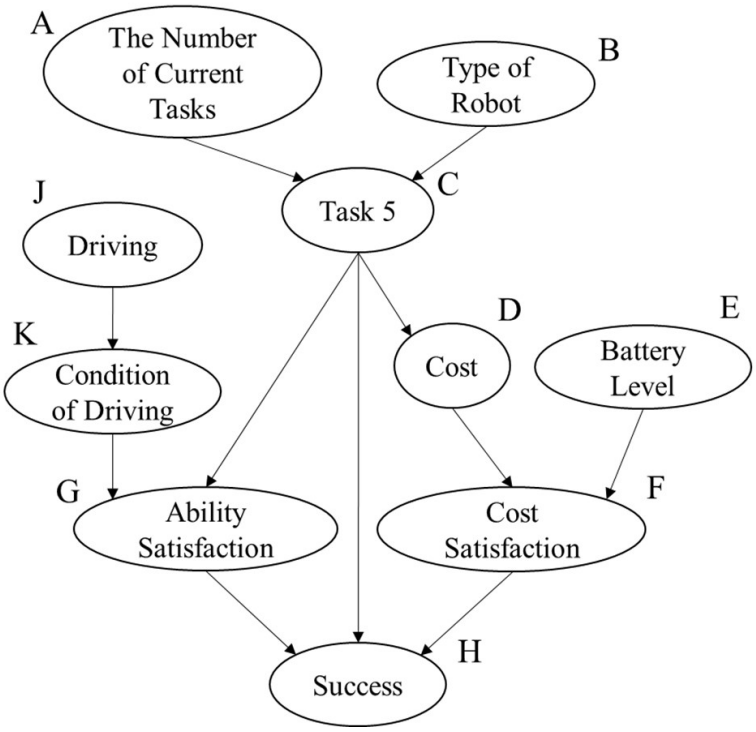

Figure 8.3. The BN of Task 5 in the Comprehensive Experiments

B	P(B)
ST	0.833333
MT	0.166667

Table 8.3. The Probability Table of Node B (in the Centralized MRS)

8.2.3 The Design of a Centralized Multi-Robot System

In the centralized MRS, Robot 1 is assigned as a central controller to deal with coordination, whereas it can still take a task in the mission. Hence, Robot 1 is an MT robot, which may coordinate robots (as one extra mission) and take a task simultaneously. The other robots are ST robots, which can only take one task concurrently. According to the number of ST and MT robots, in this case, the probability of Node B is shown in Table 8.3. The frameworks installed on the robots are illustrated in Chapter 7 as shown in Fig. 7.2 and Fig. 7.3.

B	P(B)
ST	0.666667
MT	0.333333

Table 8.4. The Probability Table of Node B (in the Decentralized MRS)

8.2.4 The Design of a Decentralized Multi-Robot System

In the decentralized MRS, Robot 1, Robot 2, and Robot 3 are in one team, and Robot 4, Robot 5, and Robot 6 are in the other team. Robot 1 and Robot 4 are assigned as team leaders to coordinate robots in a team and between teams (as one extra mission), whereas they still have a chance to take a task in the mission. Accordingly, Robot 1 and Robot 4 are MT robots, and the other robots are ST robots. Based on the number of ST and MT robots, in this situation, the probability of Node B is shown in Table 8.4. The frameworks for the robots are presented in Chapter 7 (Fig. 7.4 and Fig. 7.5).

8.2.5 The Design of a Distributed Multi-Robot System

In the distributed MRS, all robots are assumed to be ST robots in that all robots can make their own decisions to complete tasks in the mission. Thus, the probability of Node B is identical to Table 6.2 in Chapter 6. The framework for each robot is the same as Fig. 7.6 in Chapter 7.

8.3 The Experiment Result in a Centralized Multi-Robot System

In the centralized MRS, when the tasks were released, Robot 1 (the central controller) began the MRTA process with our approach. After Task 1 and Task 3 were released, Robot 3 took Task 1 and Robot 6 took Task 3 according to the collected success rates in

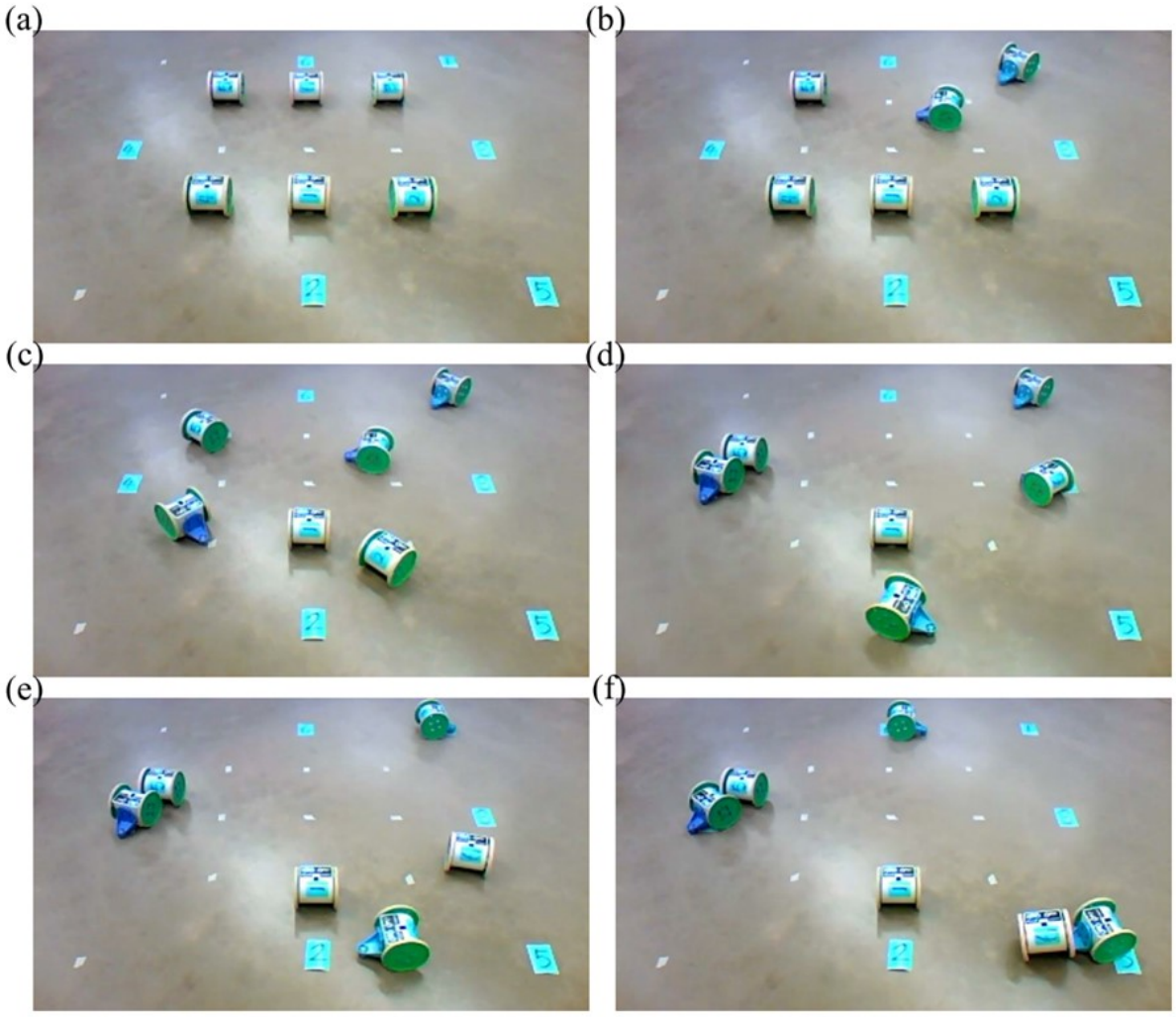

Figure 8.4. The Experiment Result in the Centralized MRS

Table 8.5. After Task 2 and Task 4 were released later, Robot 2 took Task 2 and Robot 4 and Robot 5 took Task 4, and Robot 3 and Robot 6 kept executing their current, taken tasks depending on Table 8.6. After Task 1, Task 2, Task 3, and Task 4 were completed and Task 5 and Task 6 were released, Robot 2 and Robot 6 took Task 5 and Robot 3 took Task 6 as shown in Table 8.7. In the end, all tasks were done successfully as shown in Fig. 8.4.

	Task 1	Task 3
Robot 1	0.376900	0.011642
Robot 2	0.577923	0.028283
Robot 3	0.662591	0.028283
Robot 4	0.012649	0.577923
Robot 5	0.017155	0.577923
Robot 6	0.020103	0.607180

Table 8.5. The Collected Success Rates in the First MRTA Process in the Centralized MRS

	Task 1	Task 2	Task 3	Task 4
Robot 1	0.376900	0.487360	0.011642	0.011642
Robot 2	0.577923	0.662591	0.028283	0.017155
Robot 3	0.688851	0.547552	0.028283	0.014710
Robot 4	0.012649	0.028283	0.577923	0.662591
Robot 5	0.017155	0.017155	0.577923	0.662591
Robot 6	0.023725	0.020103	0.635384	0.607180

Table 8.6. The Collected Success Rates in the Second MRTA Process in the Centralized MRS

	Task 5	Task 6
Robot 1	0.454785	0.432889
Robot 2	0.644703	0.618503
Robot 3	0.590862	0.669572
Robot 4	0.530752	0.644703
Robot 5	0.530752	0.644703
Robot 6	0.669572	0.644703

Table 8.7. The Collected Success Rates in the Third MRTA Process in the Centralized MRS

8.4 The Experiment Result in a Decentralized Multi-Robot System

In the decentralized MRS, Robot 1 and Robot 4 are coordinators, so they may start MRTA processes. The result of this experiment is shown in Fig. 8.5, and the tables of collected success rates in each MRTA process are presented in Table 8.8, Table 8.9, and Table 8.10. Although the general result is similar to the result in the centralized MRS, we may notice that the collected success rates from Robot 4 are smaller than before because Robot 4 is an MT robot and has one task (a coordination task) already in this case.

	Task 1	**Task 3**
Robot 1	0.376900	0.011642
Robot 2	0.577923	0.028283
Robot 3	0.662591	0.028283
Robot 4	0.008131	0.400135
Robot 5	0.017155	0.577923
Robot 6	0.020103	0.607180

Table 8.8. The Collected Success Rates in the First MRTA Process in the Decentralized MRS

	Task 1	**Task 2**	**Task 3**	**Task 4**
Robot 1	0.376900	0.487360	0.011642	0.011642
Robot 2	0.577923	0.662591	0.028283	0.017155
Robot 3	0.688851	0.547552	0.028283	0.014710
Robot 4	0.008131	0.014772	0.400135	0.466368
Robot 5	0.017155	0.017155	0.577923	0.662591
Robot 6	0.023725	0.020103	0.635384	0.607180

Table 8.9. The Collected Success Rates in the Second MRTA Process in the Decentralized MRS

	Task 5	Task 6
Robot 1	0.454785	0.432889
Robot 2	0.644703	0.618503
Robot 3	0.590862	0.669572
Robot 4	0.361988	0.454785
Robot 5	0.530752	0.644703
Robot 6	0.669572	0.644703

Table 8.10. The Collected Success Rates in the Third MRTA Process in the Decentralized MRS

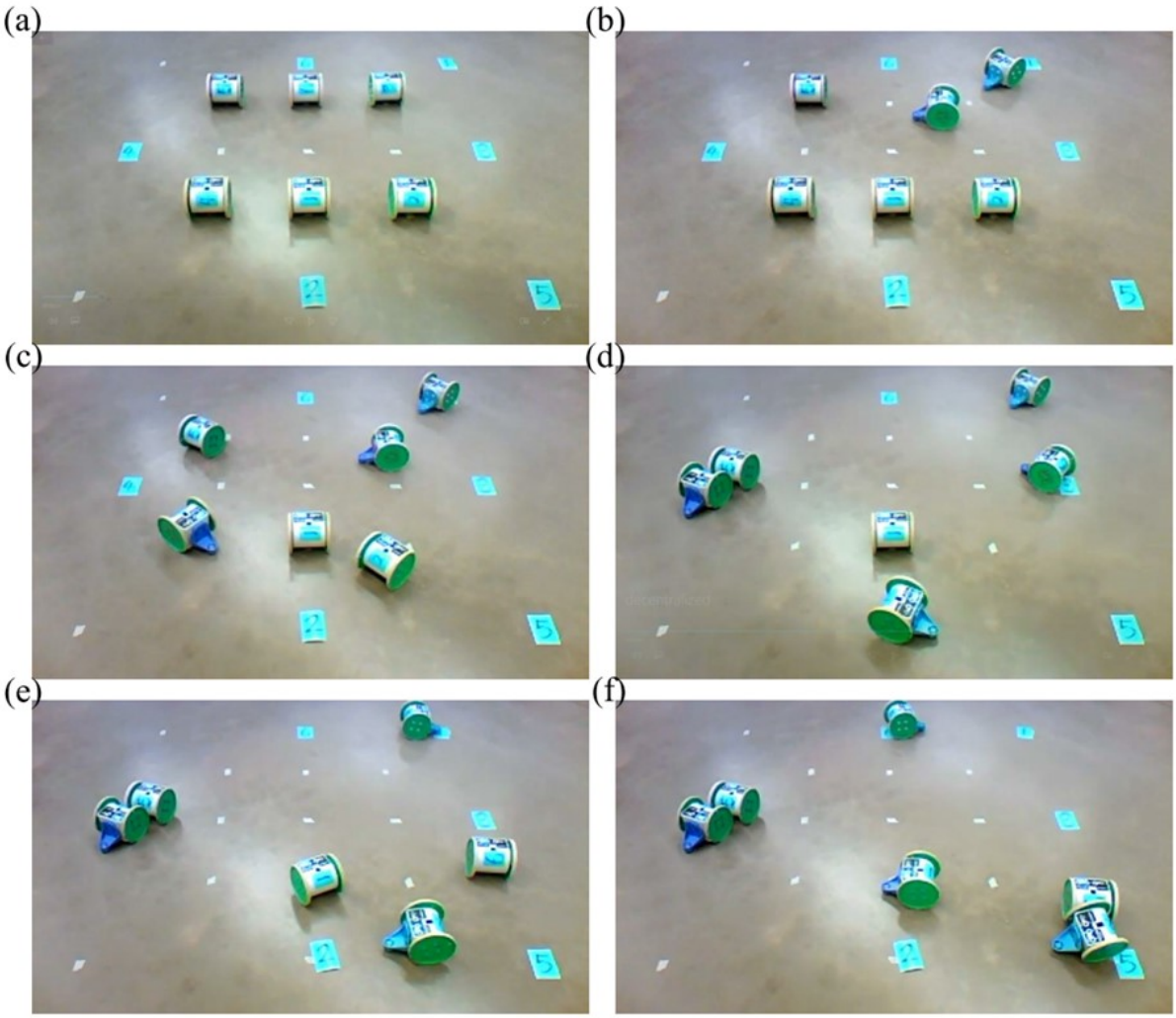

Figure 8.5. The Experiment Result in the Decentralized MRS

Robot 4 was still able to take Task 4 during the mission. Therefore, the type of robot and the number of current, taken tasks might influence MRTA results with our approach.

8.5 The Experiment Result in a Distributed Multi-Robot System

In the distributed MRS, all robots may begin MRTA processes. The overall result is shown in Fig. 8.6, which is different from the previous results. After Task 1 and Task 3 were released, Robot 3 took Task 1 and Robot 6 took Task 3 based on Table 8.11. After

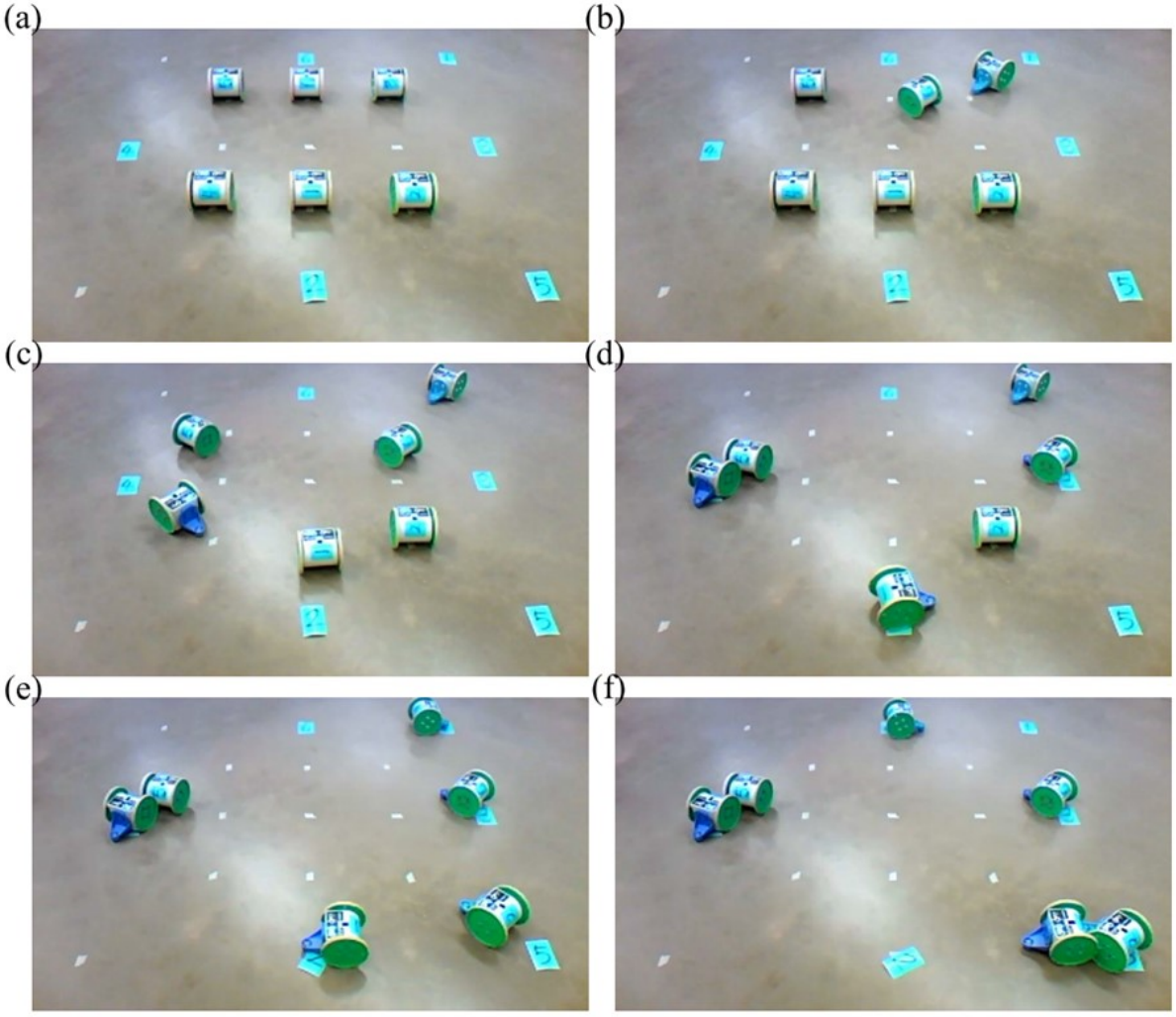

Figure 8.6. The Experiment Result in the Distributed MRS

	Task 1	Task 3
Robot 1	0.547552	0.020103
Robot 2	0.577923	0.028283
Robot 3	0.662591	0.028283
Robot 4	0.012649	0.577923
Robot 5	0.017155	0.577923
Robot 6	0.020103	0.607180

Table 8.11. The Collected Success Rates in the First MRTA Process in the Distributed MRS

	Task 1	Task 2	Task 3	Task 4
Robot 1	0.547552	0.688851	0.020103	0.020103
Robot 2	0.577923	0.662591	0.028283	0.017155
Robot 3	0.688851	0.547552	0.028283	0.014710
Robot 4	0.012649	0.028283	0.577923	0.662591
Robot 5	0.017155	0.017155	0.577923	0.662591
Robot 6	0.023725	0.020103	0.635384	0.607180

Table 8.12. The Collected Success Rates in the Second MRTA Process in the Distributed MRS

	Task 5	Task 6
Robot 1	0.669572	0.590862
Robot 2	0.693208	0.618503
Robot 3	0.590862	0.669572
Robot 4	0.530752	0.644703
Robot 5	0.530752	0.644703
Robot 6	0.669572	0.644703

Table 8.13. The Collected Success Rates in the Third MRTA Process in the Distributed MRS

Task 2 and Task 4 were released later, Robot 1 took Task 2 and Robot 4 and Robot 5 took Task 4, and Robot 3 and Robot 6 kept executing their current, taken tasks according to Table 8.12. After Task 1, Task 2, Task 3, and Task 4 were completed and Task 5 and Task 6 were released, Robot 1 and Robot 2 took Task 5 and Robot 3 took Task 6 as listed in Table 8.13. In the last MRTA process, Robot 1 and Robot 2 were chosen to take Task 5 depending on programming although the success rates from Robot 1 and Robot 6 are the same. It is possible that Robot 2 and Robot 6 take Task 5 if dissimilar programs are written. This situation might be rare when the environment is more complex, when the skill conditions vary, when the number of robots increases, when the number of values in Node D or Node E enlarges, or after training and updating CPTs. More results in a distributed MRS can be seen in case studies in Chapter 9 and Chapter 10.

8.6 The Experiment Result with an Abnormal Condition

In this abnormal situation, the MRS is designed as a distributed system because it is crucial that robots may freely communicate with each other and make their own decisions, and the mission might be fulfilled no matter whether any robots fail. Hence, all robots are ST robots. In this experiment, Robot 6 was forced to fail during the mission as a dynamic factor, whereas other robots were able to successfully complete the mission as shown in Fig. 8.7. When Robot 6 failed, it abandoned the current task, Task 3. Subsequently, Robot 4 took over Task 3 from Robot 6 according to Table 8.14. In the end, all tasks were accomplished even though the order of task execution was different from the previous results for the reason that the maximum sum of the average success rate

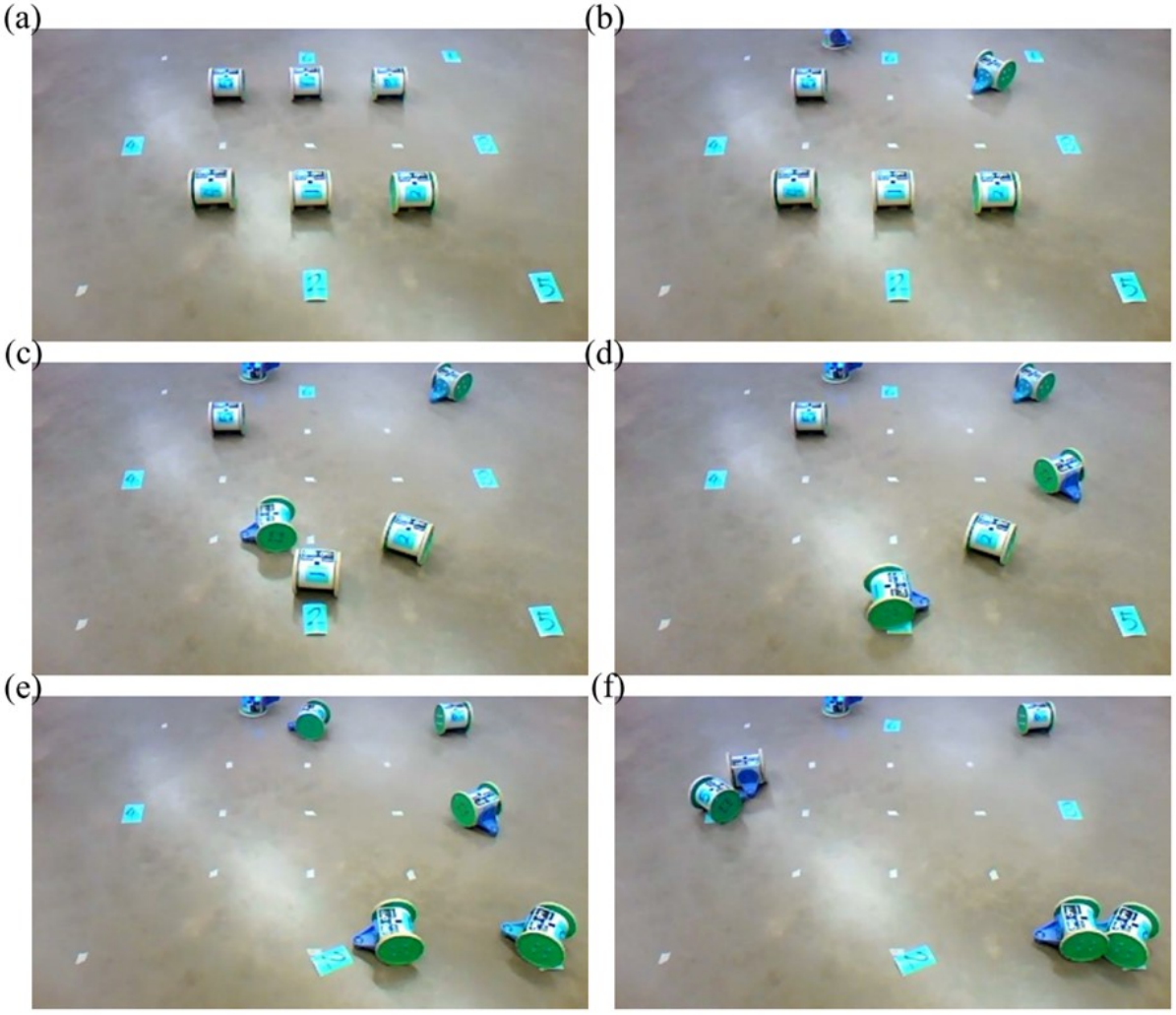

Figure 8.7. The Experiment Result with the Abnormal Condition

and the priority of tasks were considered in each MRTA process according to Table 8.15 and Table 8.16. This result correspondingly explains that our approach may overcome unpredictable surroundings.

	Task 1	Task 2	Task 3	Task 4
Robot 1	0.547552	0.688851	0.020103	0.020103
Robot 2	0.577923	0.662591	0.028283	0.017155
Robot 3	0.688851	0.547552	0.028283	0.014710
Robot 4	0.012649	0.028283	0.577923	0.662591
Robot 5	0.017155	0.017155	0.577923	0.662591
Robot 6	0.021150	0.017927	0.024046	0.020375

Table 8.14. The Collected Success Rates After Robot 6 Failed

	Task 3	Task 4	Task 5	Task 6
Robot 1	0.020103	0.020103	0.669572	0.590862
Robot 2	0.028283	0.017155	0.693208	0.618503
Robot 3	0.023725	0.010888	0.590862	0.669572
Robot 4	0.714215	0.547552	0.669572	0.644703
Robot 5	0.577923	0.662591	0.561657	0.693208
Robot 6	0.024046	0.020375	0.017618	0.029050

Table 8.15. The Collected Success Rates After Task 5 and Task 6 Were Released

	Task 4	Task 5
Robot 1	0.014710	0.715701
Robot 2	0.010888	0.737133
Robot 3	0.010888	0.590862
Robot 4	0.547552	0.669572
Robot 5	0.607180	0.530752
Robot 6	0.020375	0.017618

Table 8.16. The Collected Success Rates After Task 3 and Task 6 Were Completed

Chapter 9

Case Study: A Search-and-Rescue Mission

In this chapter, we are going to apply our approach to a real search-and-rescue mission. Our method, in the comprehensive experiments, has been validated to solve diverse types of MRTA problems (ST/MT-SR/MR-IA) in various MRSs (centralized, decentralized, and distributed) under dynamic conditions (a changeable task list and a failing robot). However, the task list of the mission was unrealistic, and the robots did not have actual abilities, which means these robots completed fabricated tasks in the mission. Consequently, an authentic search-and-rescue mission with several tasks is designed, and robots with equipment accomplish meaningful tasks in the mission. The videos of the search-and-rescue mission can be found in [169].

9.1 Problem Statements

In a realistic search-and-rescue mission, robots would be assigned to search for targets like injured people in a wreck. After locating all targets, robots for rescue would move the targets to safe places, and redundant robots in the wreck may assemble at a gathering

	Task Type	Required Skills	Target Point	Details	Priority	Released Time
Task 1	SR	Driving, Camera, Sensor	(20, 35)	Detect a target	High	First
Task 2	SR	Driving, Arm, Sensor	(20, 35)	Move the target to (2, 35)	Medium	Released when the target exists in Task 1
Task 3	SR	Driving, Camera, Sensor	(31, 20)	Detect a target	High	First
Task 4	SR	Driving, Arm, Sensor	(31, 20)	Move the target to (50, 2)	Medium	Released when the target exists in Task 3
Task 5	SR	Driving, Camera, Sensor	(45, 35)	Detect a target	High	First
Task 6	SR	Driving, Arm, Sensor	(45, 35)	Move the target to (28, 53)	Medium	Released when the target exists in Task 5
Task 7	MR	Driving, Camera	(35, 2)	None	Low	First

Table 9.1. The Task List of the Search-and-Rescue Mission

	Robot Type	Skills	Battery Level	Initial Position	Driving Speed
Robot 1	ST	Driving, Camera, Sensor	3.3 V	(10, 5)	2 in/s
Robot 2	ST	Driving, Camera, Sensor	3.1 V	(55, 5)	2 in/s
Robot 3	ST	Driving, Camera, Sensor	3.2 V	(60, 20)	2 in/s
Robot 4	ST	Driving, Arm, Sensor	3.3 V	(20, 50)	2 in/s
Robot 5	ST	Driving, Arm, Sensor	3.1 V	(5, 15)	2 in/s
Robot 6	ST	Driving, Arm, Sensor	3.2 V	(60, 45)	2 in/s

Table 9.2. The Robot List of the Search-and-Rescue Mission

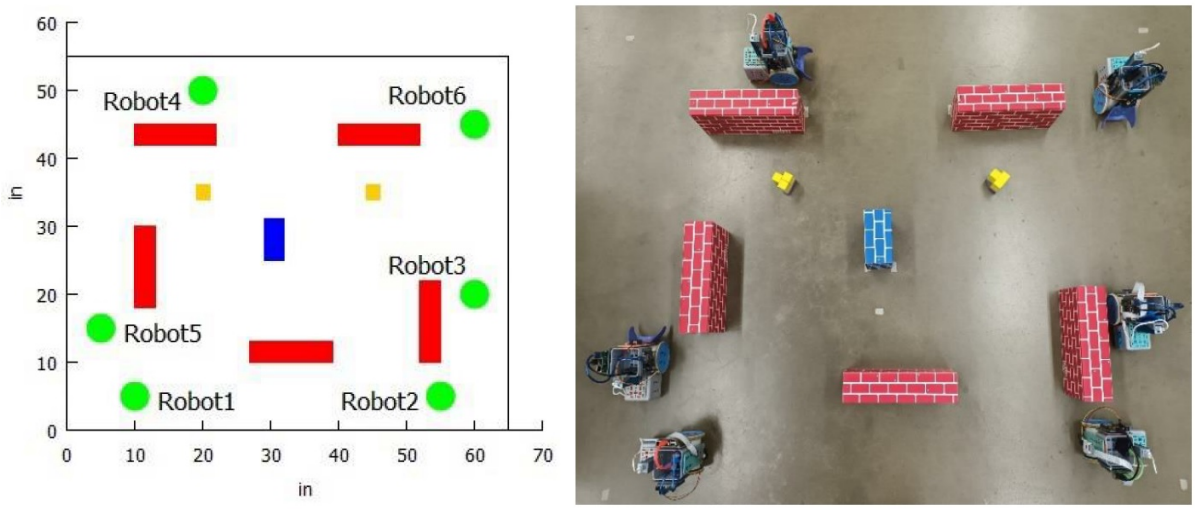

Figure 9.1. The Map of the Environment: Drawing (left) and Picture (right)

place out of the wreck. After all the targets are saved, this mission is accomplished. Therefore, based on this idea, the task list of the search-and-rescue mission is created in Table 9.1. Task 1, Task 3, and Task 5 require a robot with a camera and a sensor to search for targets at specific locations. Task 2, Task 4, and Task 6 request a robot with an arm and a sensor to move targets to safe places. Task 7 asks all working robots with cameras inside the search area to retreat to a gathering point. Hence, Task 1, Task 3, and Task 5 are released at the beginning with high priority, and Task 2, Task 4, and Task 6 with medium priority will be released when targets are found. Task 7 is released at the beginning but with low priority. As can be seen, these tasks in the search-and-rescue mission are designed for concrete purposes. The robots with cameras, arms, or sensors are listed in Table 9.2. They are located at different initial positions outside the search area. In this case, all robots are assumed to be ST robots because there is only one mission. Regarding the map in Fig. 9.1, many barriers are in the search area. The size of the red obstacle is 12" x 3", and the size of the blue obstacle is 6" x 3". The real targets are

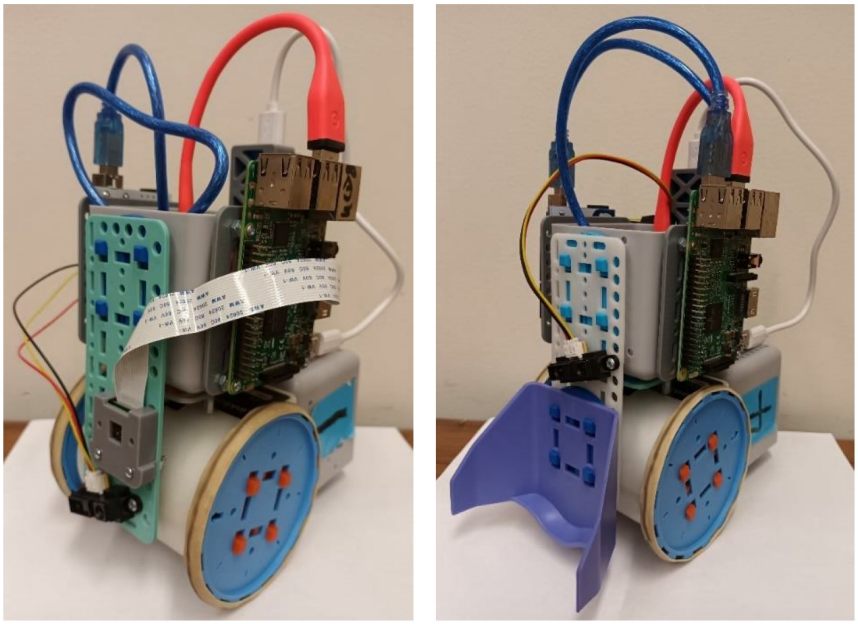

Figure 9.2. The Robot with a Camera Skill (left) and the Robot with an Arm Skill (right)

yellow. The environment is dynamic, which means the task list is changeable, robots may fail, and new obstacles may appear. Depending on this information, the MRTA problem is a heterogeneous SR/MR-ST-IA problem under dynamic conditions.

9.2 Experiment Setup

9.2.1 Hardware and Software

Each hardware, low-cost robot is composed of a Raspberry Pi [2], a Linkbot [1], an Arduino Uno, and an IR range sensor. A robot with a camera skill has an additional Pi camera, and a robot with an arm skill has an extra fixed arm as shown in Fig. 9.2. An Arduino Uno [5], which is a micro-controller, is used to receive signals from an IR range sensor and send data to a Raspberry Pi. Mobile-C [3] is utilized to build agent-based frameworks for the MRS. An OpenCV library [6] is applied to target recognition from the picture taken

[5]Arduino Official Website: https://www.arduino.cc/
[6]OpenCV Official Website: https://opencv.org/

with a Pi camera. An A* algorithm is exploited to find the shortest path from the current location to a target point.

9.2.2 The Design of a Multi-Robot System

In this case, because all robots are ST robots and the environment is dynamic, the MRS is designed as distributed to allow robots to individually make decisions. The agent-based framework is shown in Fig. 7.7 and Fig. 7.8 in Chapter 7, where robots with the same abilities utilize the identical framework.

9.2.3 The Bayesian Networks of the Tasks

According to the information about the tasks and our approach described in Chapter 6, the BNs of the tasks can be generated. Task 1, Task 3, and Task 5 require three skills (driving, camera, and sensor), so the BNs are created as Fig. 9.3 but with different task

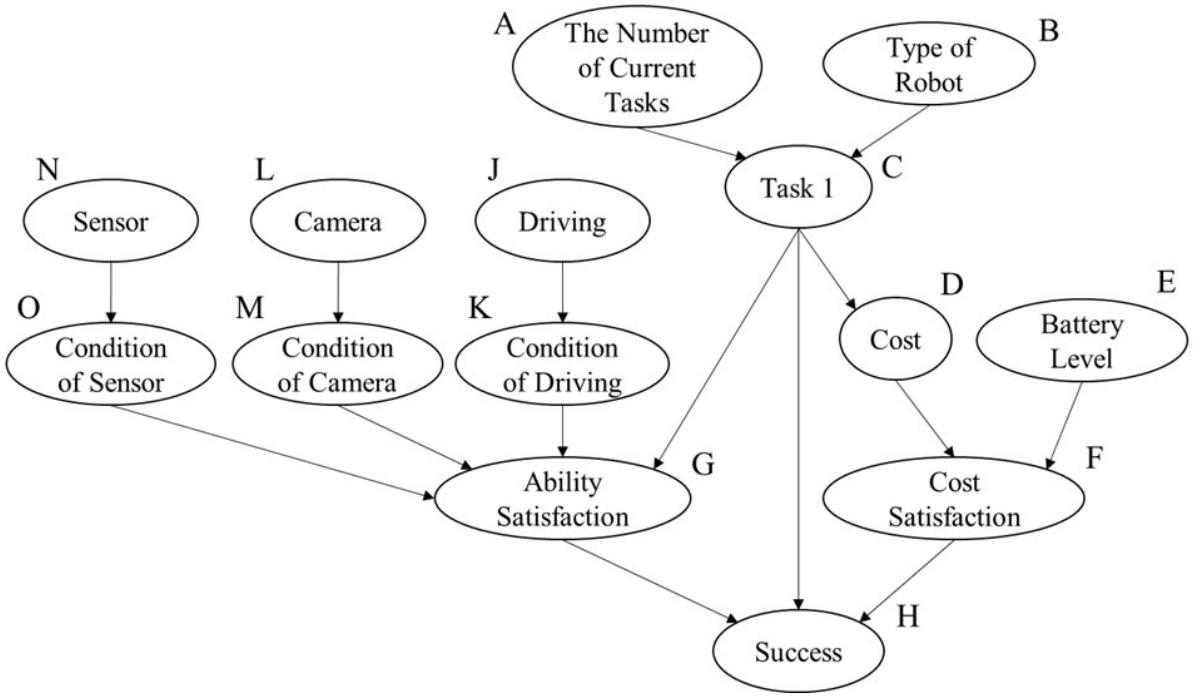

Figure 9.3. The BN of Task 1 in the Search-and-Rescue Mission

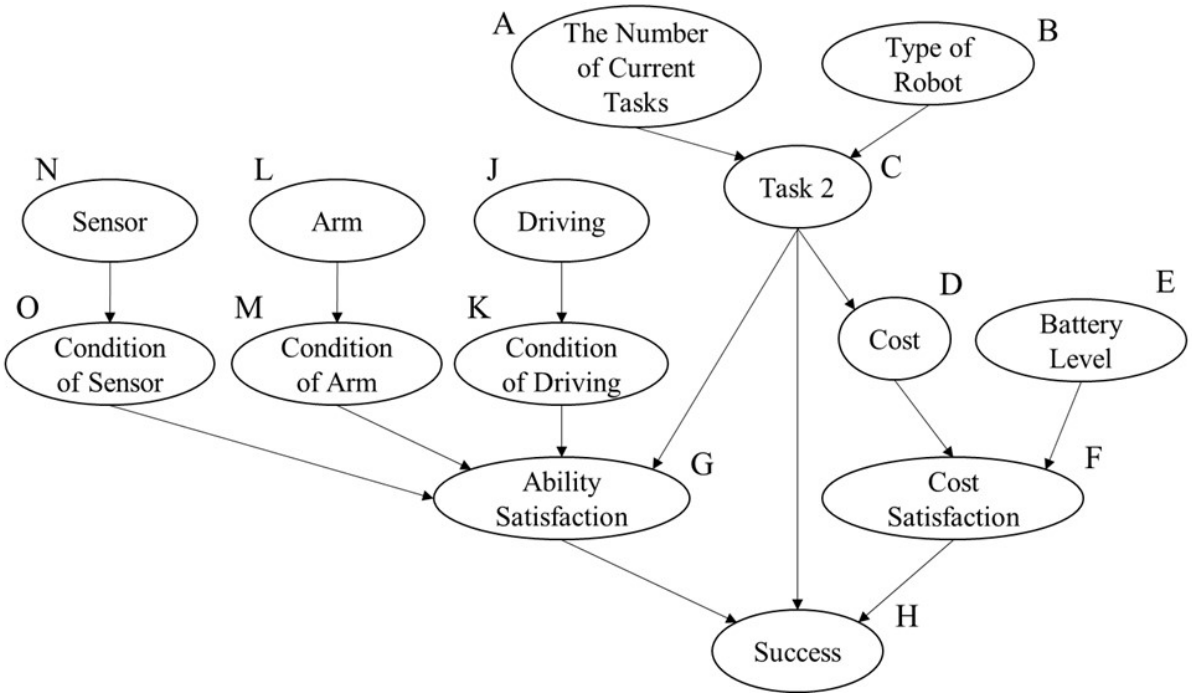

Figure 9.4. The BN of Task 2 in the Search-and-Rescue Mission

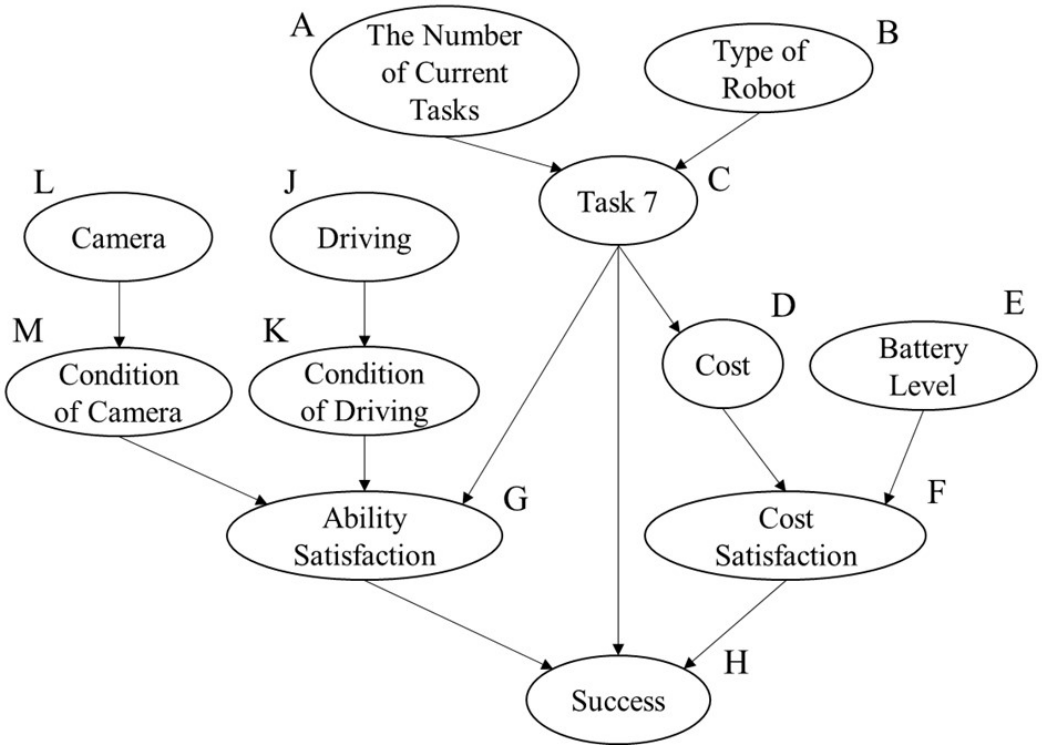

Figure 9.5. The BN of Task 7 in the Search-and-Rescue Mission

names. Task 2, Task 4, and Task 6 need three skills (driving, arm, and sensor), so the BNs are built as Fig. 9.4 except for the names of the tasks. Task 7 requests two skills (driving and camera), and its BN is shown in Fig. 9.5. The initial CPTs of the nodes in the BNs are the same as the CPTs illustrated in Chapter 6. The cost of Task 1, Task 3, or Task 5 is in seconds which can be calculated when a robot drives from the current location to the target point and recognizes the target within three seconds. The cost of Task 2, Task 4, or Task 6 is in seconds which may be calculated when a robot drives from the current location to the target point and moves the target to the safe point. The cost of Task 7 is calculated in the seconds a robot spends from the current location to the gathering point. Last, the value of the threshold of the success rate is 0.05 during the MRTA process.

9.3 The Experiment Result without Accidents

The result of the search-and-rescue mission is shown in Fig. 9.6. When the task list was released, Robot 1 took Task 1, Robot 2 took Task 3, and Robot 3 took Task 5 according to Table 9.3. Robots were not available to execute Task 7 due to low success rates and low priority. After Task 5 was completed, Task 6 was released because the target was localized. Robot 6 took Task 6 based on Table 9.4. After Task 1 and Task 3 were completed, only Task 2 was released because the target in Task 1 was found, whereas the target in Task 3 was not found. According to Table 9.5, Robot 4 took Task 2, and Robot 1, Robot 2, and Robot 3 executed Task 7. Finally, the mission was successfully completed. This result can be seen as the outcome under static surroundings since no robot fails or no unknown obstacle exists except the changeable task list.

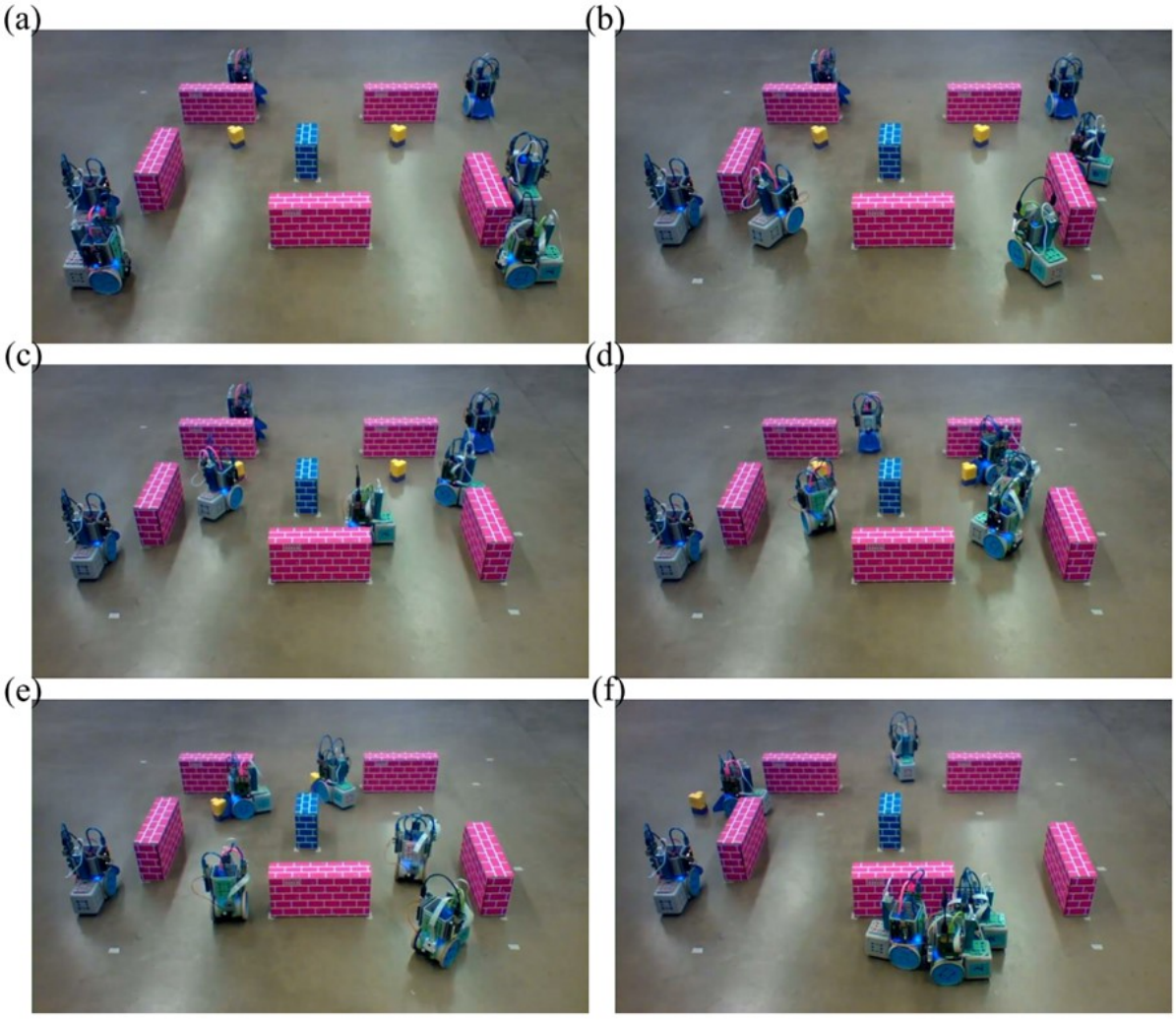

Figure 9.6. The Experiment Result without Accidents

	Task 1	**Task 3**	**Task 5**	**Task 7**
Robot 1	0.506464	0.538796	0.405057	0.607180
Robot 2	0.369700	0.506464	0.473413	0.635384
Robot 3	0.405057	0.473413	0.538796	0.547552
Robot 4	0.016968	0.010772	0.014550	0.006867
Robot 5	0.014550	0.014550	0.007954	0.017155
Robot 6	0.010772	0.012513	0.019882	0.009366

Table 9.3. The Collected Success Rates in the First MRTA Process

	Task 1	Task 3	Task 6	Task 7
Robot 1	0.631707	0.601396	0.009267	0.547552
Robot 2	0.473413	0.601396	0.012513	0.607180
Robot 3	0.506464	0.570433	0.016968	0.547552
Robot 4	0.016968	0.010772	0.369700	0.006867
Robot 5	0.014550	0.014550	0.258576	0.017155
Robot 6	0.010772	0.012513	0.439620	0.009366

Table 9.4. The Collected Success Rates After Task 5 Was Completed

	Task 2	Task 6	Task 7
Robot 1	0.023464	0.009267	0.547552
Robot 2	0.012513	0.012513	0.577923
Robot 3	0.010772	0.016968	0.547552
Robot 4	0.473413	0.369700	0.006867
Robot 5	0.439620	0.258576	0.017155
Robot 6	0.369700	0.473413	0.010888

Table 9.5. The Collected Success Rates After Task 1 and Task 3 Were Completed

9.4 The Experiment Result with One Failing Robot

In this experiment, Robot 2 was forced to fail during the mission while other robots were still able to take over tasks and successfully complete the mission as shown in Fig. 9.7. When Robot 2 failed, Robot 1 took over Task 3 based on Table 9.6. After Task 3 and Task 5 were completed, only Task 6 was released because the target in Task 5 was localized and no target was found in Task 3. Additionally, Robot 1 took Task 1, and Robot 6 took Task 6 as shown in Table 9.7. Finally, after Task 1 was completed and the target was found, Task 2 was released. Robot 4 took Task 2, and Robot 1 and Robot 3 took Task 7 according to Table 9.8. Due to the failure of Robot 2, only two robots, Robot 1

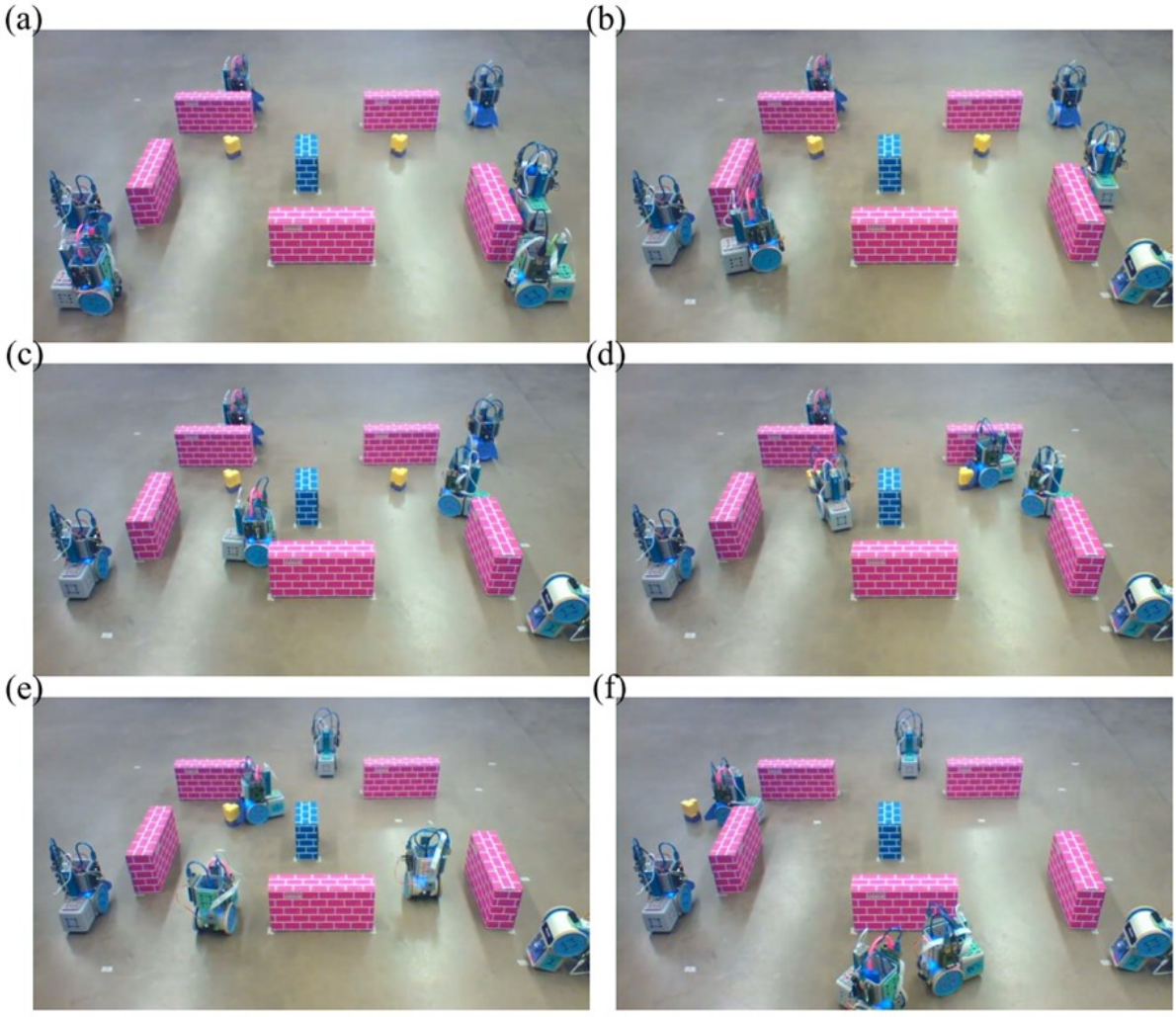

Figure 9.7. The Experiment Result with One Failing Robot

and Robot 3, collaboratively completed Task 7. In the end, all tasks were successfully accomplished even though one robot failed in the mission. This result, again, confirms that our approach may succeed under volatile surroundings when robots fail.

	Task 1	Task 3	Task 5	Task 7
Robot 1	0.538796	0.570433	0.439620	0.635384
Robot 2	0.008047	0.014729	0.014729	0.024046
Robot 3	0.439620	0.506464	0.570433	0.516003
Robot 4	0.016968	0.010772	0.014550	0.006867
Robot 5	0.014550	0.014550	0.007954	0.017155
Robot 6	0.010772	0.012513	0.019882	0.009366

Table 9.6. The Collected Success Rates After Robot 2 Failed

	Task 1	Task 6	Task 7
Robot 1	0.601396	0.009267	0.577923
Robot 2	0.008047	0.007206	0.024046
Robot 3	0.506464	0.016968	0.547552
Robot 4	0.016968	0.369700	0.006867
Robot 5	0.014550	0.258576	0.017155
Robot 6	0.010772	0.439620	0.009366

Table 9.7. The Collected Success Rates After Task 3 and Task 5 Were Completed

	Task 2	Task 6	Task 7
Robot 1	0.023464	0.009267	0.516003
Robot 2	0.005250	0.007206	0.024046
Robot 3	0.010772	0.016968	0.547552
Robot 4	0.473413	0.369700	0.006867
Robot 5	0.439620	0.258576	0.017155
Robot 6	0.473413	0.570433	0.014710

Table 9.8. The Collected Success Rates After Task 1 Was Completed

9.5 The Experiment Result with One Unknown Obstacle

In this experiment, one unknown obstacle was added to the map to simulate a dynamic environment, whereas robots may perceive it and update the map through communication. In the beginning, Robot 1 took Task 1, Robot 2 took Task 3, and Robot 3 took Task 5 as usual. However, the situation changed after the obstacle was located and Task 5 was completed. According to Table 9.9, Robot 3 took over Task 3 because the cost (the length of the route) of Task 3 for Robot 2 increased. The success rate of Task 3 for Robot 2 was

	Task 1	**Task 3**	**Task 6**	**Task 7**
Robot 1	0.631707	0.601396	0.007954	0.547552
Robot 2	0.333520	0.369700	0.004031	0.635384
Robot 3	0.506464	0.538796	0.016968	0.449083
Robot 4	0.016968	0.010772	0.369700	0.006867
Robot 5	0.014550	0.014550	0.258576	0.017155
Robot 6	0.010772	0.012513	0.439620	0.008037

Table 9.9. The Collected Success Rates After Finding an Obstacle and Completing Task 5

	Task 2	**Task 3**	**Task 6**	**Task 7**
Robot 1	0.023464	0.570433	0.009267	0.516002
Robot 2	0.004854	0.369700	0.004855	0.662590
Robot 3	0.010772	0.538796	0.016968	0.449083
Robot 4	0.473413	0.010772	0.369700	0.006867
Robot 5	0.439620	0.014550	0.258576	0.017155
Robot 6	0.405057	0.014550	0.473413	0.008037

Table 9.10. The Collected Success Rates After Task 1 Was Completed

	Task 2	Task 6	Task 7
Robot 1	0.014550	0.009267	0.547552
Robot 2	0.004854	0.004855	0.662590
Robot 3	0.010772	0.016968	0.449083
Robot 4	0.538796	0.439620	0.009366
Robot 5	0.439620	0.258576	0.017155
Robot 6	0.506464	0.570433	0.008037

Table 9.11. The Collected Success Rates After Task 3 Was Completed

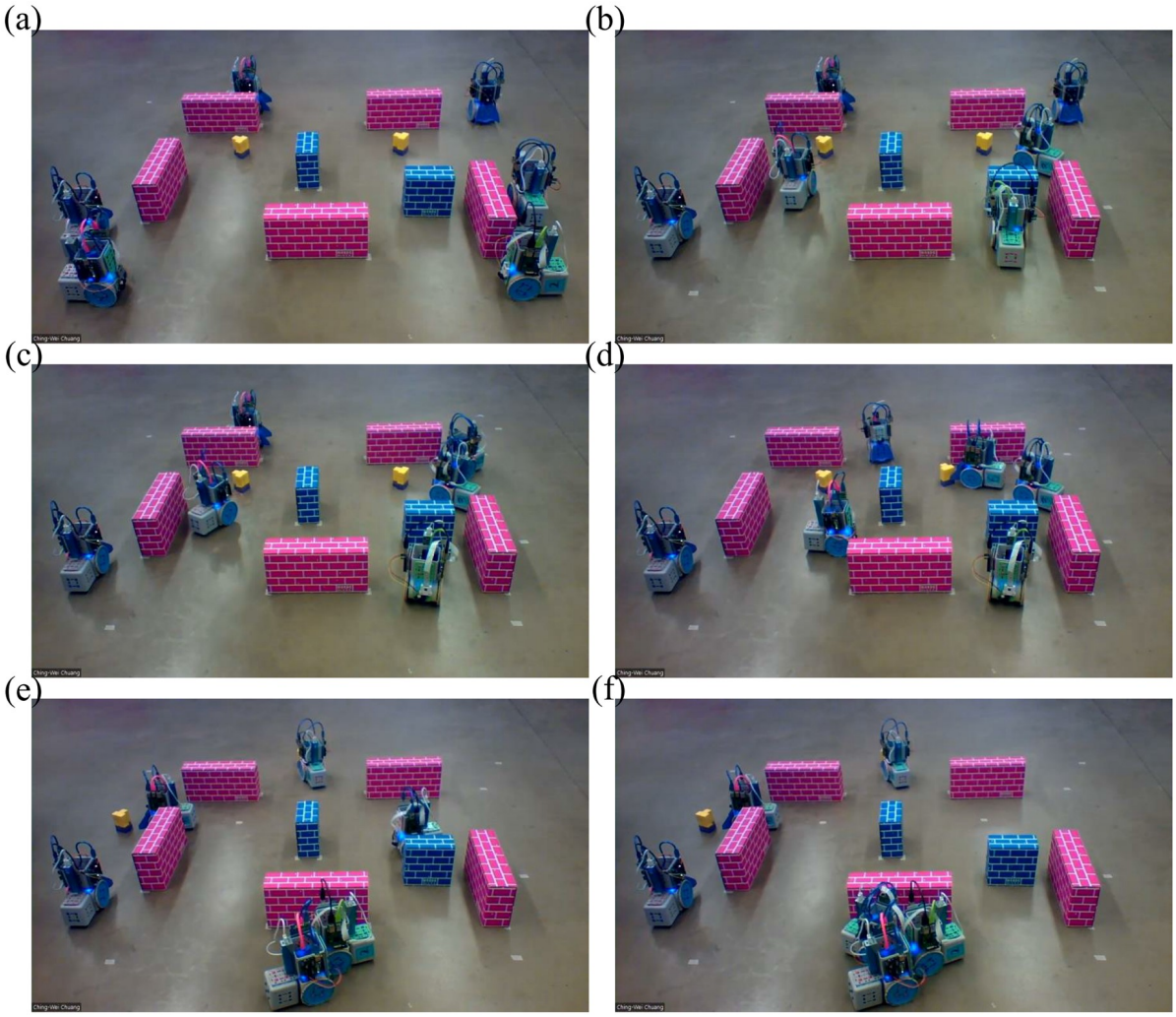

Figure 9.8. The Experiment Result with One Unknown Obstacle

smaller than the counterpart of Robot 1 or Robot 3. Additionally, Robot 6 took Task 6 which was released after the target was found in Task 5. After Task 1 was completed, Robot 1 took over Task 3, and Robot 4 took Task 2 since the target was detected in Task 1 depending on Table 9.10. Finally, after Task 3 was completed, Robot 1, Robot 2, and Robot 3 executed Task 7, and Robot 4 and Robot 6 kept executing Task 2 and Task 6 respectively based on Table 9.11. In the end, all tasks were successfully accomplished although one previously unknown obstacle was on the map as shown in Fig. 9.8. This result verifies that our approach may work in dynamic surroundings even when unknown obstacles appear.

Chapter 10

Case Study: The Training of The Search-and-Rescue Mission

In Chapter 8 and Chapter 9, only initial CPTs were utilized to solve MRTA problems although the results were rational and acceptable. We, nevertheless, could collect data from repeating experiments and update CPTs to receive more accurate success rates in order to obtain near-optimal solutions. Hence, in this chapter, we are going to collect massive data, train the CPTs of the BNs, re-complete the search-and-rescue mission, and compare the results before and after training. The video of the search-and-rescue mission after training can be found in [169].

10.1 Data Collection

In order to collect scattering data for training, robots with different voltages were normally separated on the map with disparate costs, and one robot was tested at one time. At the beginning of the execution of a task, we may know the number of current, taken tasks (Node A), the type of robot (Node B), the cost (Node D), the remaining energy (Node E), and the possessing skills (Node J, Node L, and Node N). On the other hand, we are not

able to obtain the values of Nodes F, G, K, M, and O during the execution. In the end, the value of Node H can be determined whether the robot successfully completed the task (by driving to the target point, detecting the target, or moving the target) without bumping into any obstacles. As a result, this is a supervised learning process with incomplete data.

After completing many experiments, the collected data can be seen in Appendix A. Even though we have collected 219 data, the number of data is not enough for training. We could continually repeat the experiments until enough data are collected. Nonetheless, it may take much time to collect a large amount of data from real experiments. To save time, two reasonable assumptions and two facts are designated in order to increase the number of data based on the collected data. First, after observing the results of the experiments, we found that the remaining energy was not very critical in this mission because the cost was too small to consume energy. In addition, the value of the remaining energy is not the main reason that causes the robot to fail, but the value of the cost is. The higher the cost is, the lower chance the task can be completed. Consequently, Assumption 1 and Assumption 2 are given. Furthermore, a robot without enough required skills for a task could not execute the task under any conditions, which is stated in Fact 1. Finally, a task could not be completed if no robot takes the task, so Fact 2 is listed. With these two assumptions and two facts, we increased the amount of data to 9039 to begin the training process. Partial examples of data in Task 1 are shown in Appendix B.

- Assumption 1: If a robot completes a task with a certain cost, all experiments of the task might be completed by the robot with the same or lower costs under any remaining energy.

- Assumption 2: If a robot does not complete a task with a certain cost, all experiments of the task might not be completed by the robot with the same or higher costs under any remaining energy.

- Fact 1: If a robot does not have enough skills, it could not complete the assigned task at any cost.

- Fact 2: If a robot does not take a task, the task could not be completed, and the cost could not be determined.

10.2 The Training Process

In the BNs of the tasks, eight CPTs (Nodes C, D, F, G, H, K, M, and O) are in Task 1, Task 2, Task 3, Task 4, Task 5, and Task 6, and seven CPTs (Nodes C, D, F, G, H, K, and M) are in Task 7. All CPTs could be updated through training. However, in this search-and-rescue mission, there are no MT robots, and there are no other missions assigned to this robot team. The CPT of Node C, thus, is not able to be updated in that the value of Node A is always 0 and the value of Node B is always ST. Moreover, if the number of collected data is large enough, the cost (Node D) would be a nearly normal distribution, so updating the CPT of Node D is minor. Lastly, the skill condition varies with time, so we are supposed to utilize certain algorithms to determine the current conditions during the mission. Hence, updating the CPTs of Nodes K, M, and O (the skill conditions) is inappropriate. As a result, we focus on updating the CPTs of Nodes F, G, and H in this mission.

To update the CPTs of Nodes F, G, and H in the BNs, the maximum likelihood (ML)

approach is utilized in training with a large data set. According to the descriptions in Sec. 6.5 and the incomplete data we collected, expectation maximization (EM) and the ML approach are applied during the training process.

10.2.1 Expectation Maximization (EM)

Based on the general equation, Eq. (6.4), we are able to calculate the conditional probability of the missing component (Nodes F, G, K, M, and O). For example, we may calculate the conditional probability of Node F in the case $\mathbf{d_{1-1}}$ in Appendix B as shown in Eq. (10.1) . Additionally, we can calculate the terms we need to use in the next step for the ML approach with Eq. (10.2) and Eq. (10.3). By repeating the process, all necessary terms can be calculated.

$$Pr\left(F = Y|\mathbf{d_{1-1}}\right) = \frac{Pr\left(F, \mathbf{d_{1-1}}\right)}{Pr\left(\mathbf{d_{1-1}}\right)} = 0.999379 \tag{10.1}$$

$$Pr\left(D = 3 \sim 6sec, E = 4 \sim 5V|\mathbf{d_{1-1}}\right) = \frac{Pr\left(DE, \mathbf{d_{1-1}}\right)}{Pr\left(\mathbf{d_{1-1}}\right)} = 1.0 \tag{10.2}$$

$$Pr\left(D = 3 \sim 6sec, E = 4 \sim 5V, F = Y|\mathbf{d_{1-1}}\right) = Pr\left(F = Y|\mathbf{d_{1-1}}\right) = 0.999379 \tag{10.3}$$

10.2.2 The Maximum Likelihood (ML) Approach

In the training process, we primarily update the CPTs of Node F, Node G, and Node H in each task during the learning process. As we mentioned in Chapter 6, the CPT of Node F is different in each task, so the number of cases (N) in Eq. (10.4) is only related to Task 1 instead of the whole data. Because the CPT of Node G is different from the

number of required skills, Eq. (10.5) is for the CPT of Node G with two required skills (in Task 7), and Eq. (10.6) is for the CPT of Node G with three required skills (in Tasks 1 to 6). Last, the CPT of Node H is the same in each task, so the number of cases (N) in Eq. (10.7) is the total number of cases in the incomplete data ($\mathbf{D}$). By writing respective equations, we are able to update the CPTs of Node F, Node G, and Node H in each task as shown in Appendix C where the maximum probability is 0.999999999999999 and the minimum probability is 0.000000000000001 to avoid the conditional probability which is zero or one during the training process.

$$\begin{aligned} P^1 &= (F = Y|D = 3 \sim 6\ sec, E = 4 \sim 5\ V) \\ &= \frac{\sum_{i=1}^{N} Pr_{P^0}(F, DE|\mathbf{d_i})}{\sum_{i=1}^{N} Pr_{P^0}(DE|\mathbf{d_i})} = 0.8747023052 \end{aligned} \tag{10.4}$$

$$\begin{aligned} P^1 &= (G = Y|C = Y, K = G, M = G) \\ &= \frac{\sum_{i=1}^{N} Pr_{P^0}(G, CKM|\mathbf{d_i})}{\sum_{i=1}^{N} Pr_{P^0}(CKM|\mathbf{d_i})} = 0.9856118613 \end{aligned} \tag{10.5}$$

$$\begin{aligned} P^1 &= (G = Y|C = Y, K = G, M = G, O = G) \\ &= \frac{\sum_{i=1}^{N} Pr_{P^0}(G, CKMO|\mathbf{d_i})}{\sum_{i=1}^{N} Pr_{P^0}(CKMO|\mathbf{d_i})} = 0.9765505897 \end{aligned} \tag{10.6}$$

$$\begin{aligned} P^1 &= (H = Y|C = Y, F = Y, G = Y) \\ &= \frac{\sum_{i=1}^{N} Pr_{P^0}(H, CFG|\mathbf{d_i})}{\sum_{i=1}^{N} Pr_{P^0}(CFG|\mathbf{d_i})} = 0.9890817593 \end{aligned} \tag{10.7}$$

After obtaining all the updated CPTs, the first iteration of the updating process is completed, whereas the result does not converge according to the ML theory. Based

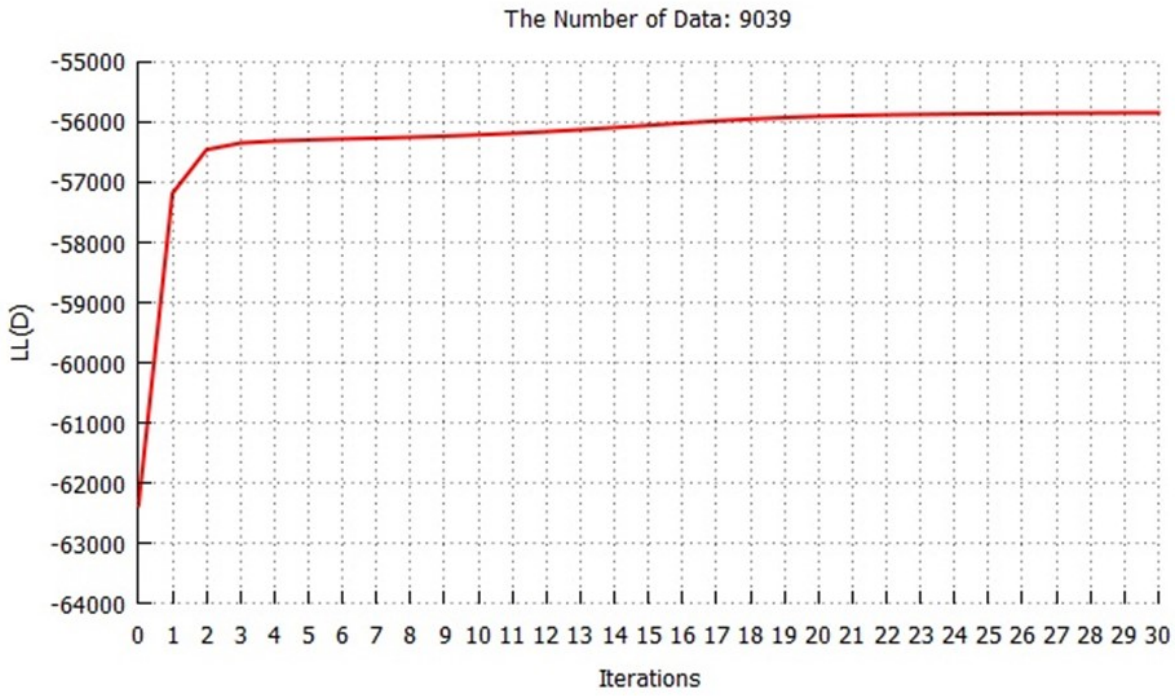

Figure 10.1. Log-Likelihood vs Iterations During Training

on Fig. 10.1, the result of the log-likelihood would converge to around -55850 after 30 iterations. We, hence, set -55860 as the value of the threshold to stop the training process and obtain the settled results of the updated CPTs of Nodes F, G, and H as shown in Tables 10.1 to 10.10.

P(F=Yes|DE)		E		
		2∼3 V	3∼4 V	4∼5 V
D	0∼3 sec	0.9999990811	0.9999994158	0.9999999840
	3∼6 sec	0.9999988613	0.9999990811	0.9999994158
	6∼9 sec	0.9999968748	0.9999972584	0.9999977927
	9∼12 sec	0.9999908872	0.9999916785	0.9999927147
	12∼15 sec	0.9999706650	0.9999726065	0.9999750336
	15∼18 sec	0.9998919410	0.9998977128	0.9999046686
	18∼21 sec	0.9874800273	0.9879035644	0.9884080828
	21∼24 sec	0.9043875314	0.9055226293	0.9068725315
	24∼27 sec	0.8755338290	0.8766120706	0.8778627080
	27∼30 sec	0.7499420854	0.7511760344	0.7525677711
	30∼33 sec	0.5259716828	0.5277492729	0.5296834505
	33∼36 sec	0.0589417835	0.0619178070	0.0649457952
	36∼39 sec	0.0004801259	0.0005950417	0.0007259993
	39∼42 sec	0.0003782633	0.0004801259	0.0005950417
	Larger than 42 sec	0.0002871695	0.0003782633	0.0004801259

Table 10.1. The Updated CPT of Node F in Task 1

P(F=Yes|DE)		E		
		2∼3 V	3∼4 V	4∼5 V
D	0∼3 sec	0.9999917000	0.9999947702	0.9999998584
	3∼6 sec	0.9999896511	0.9999917000	0.9999947702
	6∼9 sec	0.9999881699	0.9999896511	0.9999917000
	9∼12 sec	0.9999870372	0.9999881699	0.9999896511
	12∼15 sec	0.9999563043	0.9999592204	0.9999628616
	15∼18 sec	0.9998296111	0.9998388130	0.9998498904
	18∼21 sec	0.9991968254	0.9992336462	0.9992766533
	21∼24 sec	0.9951895039	0.9953836717	0.9956047707
	24∼27 sec	0.8838514680	0.8862264714	0.8889289949
	27∼30 sec	0.4165085183	0.4189662479	0.4216517147
	30∼33 sec	0.4142253339	0.4165085183	0.4189662479
	33∼36 sec	0.0003614161	0.0004411997	0.0005332580
	36∼39 sec	0.0000785890	0.0000975790	0.0001192799
	39∼42 sec	0.0000208068	0.0000264835	0.0000329160
	Larger than 42 sec	0.0000157532	0.0000208068	0.0000264835

Table 10.2. The Updated CPT of Node F in Task 2

P(F=Yes|DE)		E		
		2∼3 V	3∼4 V	4∼5 V
D	0∼3 sec	0.9999958237	0.9999973607	0.9999999282
	3∼6 sec	0.9999948035	0.9999958237	0.9999973607
	6∼9 sec	0.9999940687	0.9999948035	0.9999958237
	9∼12 sec	0.9999812980	0.9999829431	0.9999850913
	12∼15 sec	0.9999337687	0.9999382289	0.9999437906
	15∼18 sec	0.9997254192	0.9997404261	0.9997584682
	18∼21 sec	0.9986029139	0.9986679128	0.9987437351
	21∼24 sec	0.9908372764	0.9912115108	0.9916372525
	24∼27 sec	0.8115032455	0.8149690067	0.8189060743
	27∼30 sec	0.4143982506	0.4193136667	0.4246955225
	30∼33 sec	0.0080687816	0.0097139483	0.0116281569
	33∼36 sec	0.0066352222	0.0080687816	0.0097139483
	36∼39 sec	0.0053711888	0.0066352222	0.0080687816
	39∼42 sec	0.0042451255	0.0053711888	0.0066352222
	Larger than 42 sec	0.0032328927	0.0042451255	0.0053711888

Table 10.3. The Updated CPT of Node F in Task 3

P(F=Yes|DE)		E		
		2∼3 V	3∼4 V	4∼5 V
D	0∼3 sec	0.9999825557	0.9999890450	0.9999997048
	3∼6 sec	0.9999781977	0.9999825557	0.9999890450
	6∼9 sec	0.9999750336	0.9999781977	0.9999825557
	9∼12 sec	0.9999726065	0.9999750336	0.9999781977
	12∼15 sec	0.9999706650	0.9999726065	0.9999750336
	15∼18 sec	0.9999690579	0.9999706650	0.9999726065
	18∼21 sec	0.9999676876	0.9999690579	0.9999706650
	21∼24 sec	0.9998827031	0.9998870155	0.9998919410
	24∼27 sec	0.9994899649	0.9995067953	0.9995255781
	27∼30 sec	0.9849074891	0.9852756184	0.9856828894
	30∼33 sec	0.8198314895	0.8210962653	0.8224950738
	33∼36 sec	0.5695481841	0.5704002269	0.5713153833
	36∼39 sec	0.5222650682	0.5229841294	0.5237381312
	39∼42 sec	0.3360512987	0.3371166031	0.3381928275
	Larger than 42 sec	0.1119797242	0.1144585996	0.1168427140

Table 10.4. The Updated CPT of Node F in Task 4

P(F=Yes|DE)		E		
		2∼3 V	3∼4 V	4∼5 V
D	0∼3 sec	0.9999990811	0.9999994158	0.9999999840
	3∼6 sec	0.9999988613	0.9999990811	0.9999994158
	6∼9 sec	0.9999968748	0.9999972584	0.9999977927
	9∼12 sec	0.9999908872	0.9999916785	0.9999927147
	12∼15 sec	0.9999706650	0.9999726065	0.9999750336
	15∼18 sec	0.9998919410	0.9998977128	0.9999046686
	18∼21 sec	0.9995255781	0.9995470243	0.9995721010
	21∼24 sec	0.9973950226	0.9974986639	0.9976167874
	24∼27 sec	0.9813547949	0.9820166619	0.9827553517
	27∼30 sec	0.5770413086	0.5807142938	0.5847836125
	30∼33 sec	0.3322044010	0.3358520523	0.3397515336
	33∼36 sec	0.0033762370	0.0041101507	0.0049537119
	36∼39 sec	0.0027301499	0.0033762370	0.0041101507
	39∼42 sec	0.0021555343	0.0027301499	0.0033762370
	Larger than 42 sec	0.0016398833	0.0021555343	0.0027301499

Table 10.5. The Updated CPT of Node F in Task 5

P(F=Yes|DE)		E		
		2∼3 V	3∼4 V	4∼5 V
D	0∼3 sec	0.9999958237	0.9999973607	0.9999999282
	3∼6 sec	0.9999948035	0.9999958237	0.9999973607
	6∼9 sec	0.9999940687	0.9999948035	0.9999958237
	9∼12 sec	0.9999935082	0.9999940687	0.9999948035
	12∼15 sec	0.9999930614	0.9999935082	0.9999940687
	15∼18 sec	0.9999926921	0.9999930614	0.9999935082
	18∼21 sec	0.9999779685	0.9999788958	0.9999799834
	21∼24 sec	0.9999241555	0.9999269173	0.9999300713
	24∼27 sec	0.9996913039	0.9997013688	0.9997125963
	27∼30 sec	0.9984481696	0.9984948890	0.9985459284
	30∼33 sec	0.8198314895	0.8210962653	0.8224950738
	33∼36 sec	0.8186621765	0.8198314895	0.8210962653
	36∼39 sec	0.5227286175	0.5243106840	0.5259716828
	39∼42 sec	0.1880215101	0.1904457521	0.1928634230
	Larger than 42 sec	0.0247508905	0.0269904427	0.0292164940

Table 10.6. The Updated CPT of Node F in Task 6

P(F=Yes|DE)		E		
		2∼3 V	3∼4 V	4∼5 V
D	0∼3 sec	0.9999999338	0.9999999576	0.9999999988
	3∼6 sec	0.9999998460	0.9999998752	0.9999999203
	6∼9 sec	0.9999996505	0.9999996923	0.9999997512
	9∼12 sec	0.9999991855	0.9999992538	0.9999993440
	12∼15 sec	0.9999979943	0.9999981206	0.9999982801
	15∼18 sec	0.9999946594	0.9999949263	0.9999952513
	18∼21 sec	0.9999842661	0.9999849182	0.9999856884
	21∼24 sec	0.9999473171	0.9999491987	0.9999513639
	24∼27 sec	0.9997926354	0.9997992499	0.9998066876
	27∼30 sec	0.9989973135	0.9990269175	0.9990595188
	30∼33 sec	0.8566112525	0.8577081940	0.8589193177
	33∼36 sec	0.7891021614	0.7900806466	0.7911376291
	36∼39 sec	0.6596248367	0.6610505019	0.6625510497
	39∼42 sec	0.2394862591	0.2427872056	0.2461138540
	Larger than 42 sec	0.0006593094	0.0008699476	0.0011059778

Table 10.7. The Updated CPT of Node F in Task 7

C	**K**	**M**	**P(G=Yes|CKM)**
Yes	Good	Good	0.999999999999999
Yes	Good	Bad	0.000000002532956
Yes	Bad	Good	0.001846195281535
Yes	Bad	Bad	0.000000000000001
No	Good	Good	0.500000
No	Good	Bad	0.500000
No	Bad	Good	0.500000
No	Bad	Bad	0.500000

Table 10.8. The Updated CPT of Node G (Two Required Skills)

C	K	M	O	P(G=Yes|CKMO)
Yes	Good	Good	Good	0.999999999999999
Yes	Good	Good	Bad	0.000239455968839
Yes	Good	Bad	Good	0.000239455968839
Yes	Good	Bad	Bad	0.000000000000001
Yes	Bad	Good	Good	0.612144892119595
Yes	Bad	Good	Bad	0.000000000000001
Yes	Bad	Bad	Good	0.000000000000001
Yes	Bad	Bad	Bad	0.000000000000001
No	Good	Good	Good	0.500000
No	Good	Good	Bad	0.500000
No	Good	Bad	Good	0.500000
No	Good	Bad	Bad	0.500000
No	Bad	Good	Good	0.500000
No	Bad	Good	Bad	0.500000
No	Bad	Bad	Good	0.500000
No	Bad	Bad	Bad	0.500000

Table 10.9. The Updated CPT of Node G (Three Required Skills)

C	F	G	P(H=Yes|CFG)
Yes	Yes	Yes	0.999999999999999
Yes	Yes	No	0.000000008771653
Yes	No	Yes	0.028478206613453
Yes	No	No	0.000000000000001
No	Yes	Yes	0.000000000000001
No	Yes	No	0.000000000000001
No	No	Yes	0.000000000000001
No	No	No	0.000000000000001

Table 10.10. The Updated CPT of Node H

10.3 The Experiment Result after Training

To compare the results of MRTA before and after training, we may re-complete the search-and-rescue mission after the updated CPTs are obtained during the training process. With the same information about the tasks and the identical robots in the mission, all tasks were successfully completed under the condition without accidents although the values of the success rates in each MRTA process were totally different. According to Table 10.11, Task 1 was taken by Robot 1, Task 3 was taken by Robot 2, and Task 5 was taken by Robot 3,

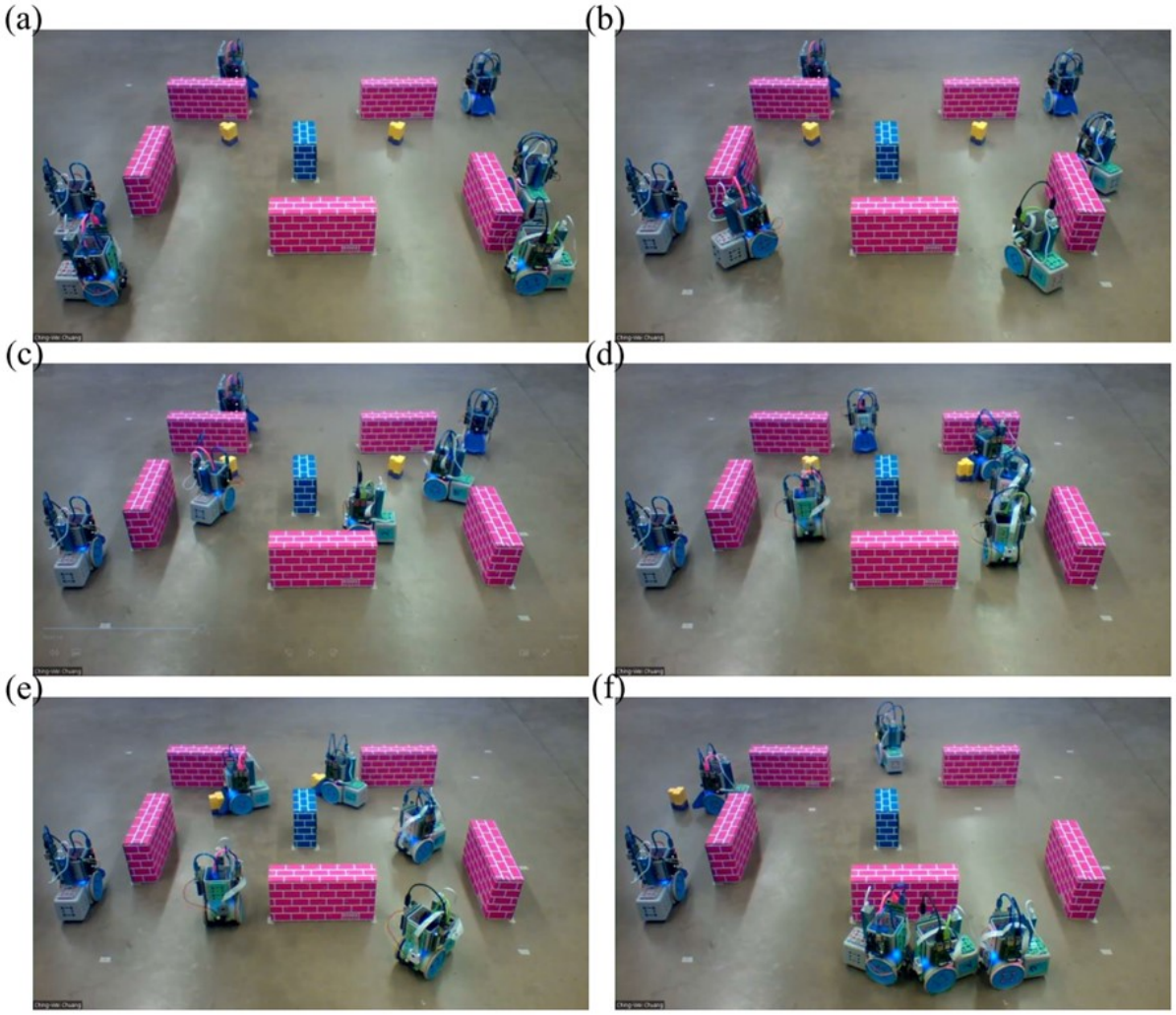

Figure 10.2. The Experiment Result without Accidents after Training

	Task 1	Task 3	Task 5	Task 7
Robot 1	0.6645787882	0.6723121732	0.3985495426	0.6482979023
Robot 2	0.3639458691	0.6716114673	0.6708475611	0.6482986160
Robot 3	0.5099174286	0.6667399711	0.6724149334	0.6482895869
Robot 4	0.0009021152	0.0007400230	0.0009018079	0.0005734345
Robot 5	0.0008916022	0.0009010373	0.0003200713	0.0007203385
Robot 6	0.0007940539	0.0008945017	0.0009021809	0.0007196610

Table 10.11. The Collected Success Rates in the First MRTA Process After Training

	Task 1	Task 3	Task 6	Task 7
Robot 1	0.6724811604	0.6724414038	0.0009008857	0.6482670895
Robot 2	0.6645787882	0.6724783657	0.0009021408	0.6482979023
Robot 3	0.6645787882	0.6724414038	0.0009021988	0.6482895869
Robot 4	0.0009021152	0.0007400230	0.5555984610	0.0005734345
Robot 5	0.0008916022	0.0009010373	0.1435751284	0.0007203385
Robot 6	0.0007940539	0.0008945017	0.6722866558	0.0007196610

Table 10.12. The Collected Success Rates After Task 5 Was Completed After Training

which was the same as the result before training. Nevertheless, the collected success rates were different. The chosen success rates in each task were higher than the values before training. In other words, the success rate would be more accurate than before because the updated CPTs were built from the collected data with real experiments. After Task 5 was completed, Task 6 was released due to the existing target, and Robot 6 took it depending on Table 10.12. It can also be observed that the success rate of Task 6 for Robot 6 was higher than before. After Task 1 and Task 3 were completed and Task 2 was released, Task 2 was taken by Robot 4, and Robot 1, Robot 2, and Robot 3 took Task 7 based on Table 10.13. The result was the same as before, whereas the success rates were also

	Task 2	Task 6	Task 7
Robot 1	0.0009021945	0.0009008857	0.6482670895
Robot 2	0.0008981586	0.0009021408	0.6482958904
Robot 3	0.0008024810	0.0009021988	0.6482895869
Robot 4	0.6694657718	0.5555984610	0.0005734345
Robot 5	0.5981500233	0.1435751284	0.0007203385
Robot 6	0.2928745807	0.6724340136	0.0007202015

Table 10.13. The Collected Success Rates After Task 1 and Task 3 Were Completed After Training

higher or more accurate than before. In the end, all tasks were accomplished as shown in Fig. 10.2. As can be seen, the MRTA results, in this case, are the same no matter whether using the initial CPTs created based on experts' knowledge or the updated CPTs built with collected data. Nonetheless, the value of the success rate is more accurate or closer to the real situation, which represents the MRTA result will be near-optimal after training with a large amount of data.

Chapter 11

Conclusion

In this dissertation, the novel approach with Bayesian Networks (BNs) in multi-robot task allocation (MRTA) and the generic agent-based framework of multi-robot systems (MRSs) are proposed and elaborated with several vivid examples and realistic experiments to demonstrate that our approach could be efficiently applied to most applications and effectively overcome current difficulties and challenges in robotics. In the beginning, we summarized the classification of MRSs, the current frameworks of MRSs, the categorization of MRTA, the existing MRTA methodology, and prevalent machine-learning methods in robotics. Due to the diversity of MRS frameworks and the challenges of MRTA approaches under dynamic environments, an innovative viewpoint to solve MRTA problems and the general structure of MRSs were introduced. The task success rate which can be calculated with the BNs of tasks and Bayes' theorem is considered in MRTA instead of mere costs and rewards. Necessary agents are organized with agencies to construct various MRSs even when state-of-the-art methods are involved in diverse applications. In the end, the comprehensive experiments and the search-and-rescue mission with low-cost hardware robots demonstrated that our proposed approach could competently solve chal-

lenging problems under miscellaneous situations. Our contributions to this research are listed and described as follows.

- A BN is used to combine the conditions of skills, the conditions of robots, and the requirement of a task. It is rare to utilize a generalized algorithm to simultaneously consider the current state of a robot and the ability condition during an MRTA process. Several algorithms may vary in different applications with existing MRTA approaches, whereas they are specifically designed. With BNs, it is simple to integrate dynamic factors in numerous applications.

- The parameter setup is formalized. Costs and rewards in the existing MRTA approach are commonly designed by experts case by case or arranged with trial and error. With BNs, CPTs can be reasonably built with experts' knowledge or with collected data from real experiments without trial and error. The parameters in CPTs may reflect authentic situations.

- The design of BNs of tasks is simple. Many algorithms of MRTA may be difficult to design, especially for behavior-based approaches. With our technique, only the nodes related to abilities are added to the general Bayesian Network with small modifications. Particularly, most CPTs are commonly used in the BNs of tasks without re-setup. Because of these reasons, it is easy to build BNs to comply with the requirements of tasks.

- The success rates of tasks are utilized to allocate tasks to robots during an MRTA process. A utility calculated with costs and rewards is not the only factor to assign

tasks to robots now that the conditions of robots and skills are needed to be considered. It is more appropriate to allocate tasks with the success rate calculated based on the BN and Bayes' theorem. Finding the maximum sum of task success rates is utilized to benefit a group of robots in static and dynamic environments.

- The proposed approach to MRTA is learnable. Most current MRTA methods are not learnable. The CPT of a BN, nevertheless, can be updated with a large data set to reflect realistic situations, which means our approach is learnable.

- The generic agent-based framework of MRSs is provided. Considering more and more modern strategies are utilized in robotics and most MRS structures are designed independently, essential agents are introduced and classified with agencies in the generic agent-based framework. With this framework, researchers may add or create needed agents to execute respective programs in their applications without changing the basic structure of MRSs.

- The new approach to MRTA and the generic agent-based framework were testified with hardware, low-cost robots under dynamic conditions. Some methods or algorithms were only in theories or examined in simulations; however, our approach was demonstrated with hardware robots. Additionally, highly efficient hardware might not be required with our model. Researchers or workers in companies with limited resources, therefore, may be able to apply our approach to their applications or products.

Chapter 12

Future Work

Through the comprehensive experiments and the search-and-rescue mission, we have already demonstrated that our approach could be efficiently applied to different types of MRSs (centralized, decentralized, and distributed), dynamic environments (adjustable task lists, changeable maps, and robot conditions), distinctive types of MRTA problems (ST or MT robots, SR or MR tasks, instantaneous assignments, and the priority of tasks), and a heterogeneous, middle-scale robot team. Nonetheless, our approach might not be limited to these applications. Researchers may extend our methodology to other fields. Possible future works and extensions are depicted as follows.

- A robot team may be homogeneous. Even though our approach is designed to overcome dynamic conditions, we may still apply it to a homogeneous robot team. The type of robot, the energy condition, the type of skill on each robot, and the skill condition can be assumed the same, so all robots would be identical, which creates a homogeneous robot team. Therefore, our approach is very applicable to a homogeneous MRS.

- The size of a robot team may increase. Only six robots were tested in our experiments; notwithstanding, the number of robots may increase to analyze scalability. Our MRTA approach is a hybrid method, so its scalability could be between the scalability with an optimization-based approach in a centralized MRS and the counterpart with an auction-based approach in a distributed MRS. In the future, hardware experiments with more robots are needed to assess scalability.

- An MRTA problem could be a time-extended-assignment (TA) problem under static environments. Due to dynamic environments, TA problems were not considered. However, because robots would not fail in static environments, it is possible to solve TA problems to minimize the execution time with the highest success rate. The optimal combination in our approach might be modified to obtain the maximum sum of success rates when including the execution order of tasks.

- Several missions are executed at the same time. There is only one mission in the search-and-rescue mission, so we are not able to observe the huge influence of ST and MT robots in a team. Hereafter, multiple missions might be concurrently assigned to a robot team to verify MT robots might take one task in each mission simultaneously.

- Other machine-learning or modern approaches may be included in an MRS. Only BNs are utilized in MRTA in this research although we have already thought of state-of-the-art methods when proposing the generic agent-based framework. Researchers may apply their modern techniques including on-board learning to fault detection,

perception, information fusion, motion planning, and coordination to validate the flexibility of our framework.

- The generic agent-based framework might work in other software frameworks. Although Mobile-C is the only software framework we used in this dissertation, the generic agent-based framework is not specialized to Mobile-C. Researchers may utilize other software frameworks to build agent-based frameworks based on the generic agent-based framework to analyze its workability.

- Our methodology may be applied to other research areas. The proposed approach was only implemented in search and rescue in that robot types and environments are usually more complicated than other applications. Researchers and scholars in robotics may apply our approach to their interesting fields, such as security patrolling, surveillance, material cleaning, warehouse systems, assembly systems, exploration, games with robots, or even robotic arms with several joints. These applications might examine the extension of our approach in the future.

Appendix A

Collected Data from Experiments

	A	B	C	D	E	H	J	L	N
d_1	0	ST	Y	5.828 s	4.1 V	Y	Y	Y	Y
d_2	0	ST	Y	8.914 s	4.1 V	Y	Y	Y	Y
d_3	0	ST	Y	11.328 s	4.1 V	Y	Y	Y	Y
d_4	0	ST	Y	14.950 s	4.1 V	Y	Y	Y	Y
d_5	0	ST	Y	15.036 s	4.1 V	Y	Y	Y	Y
d_6	0	ST	Y	20.243 s	4.1 V	N	Y	Y	Y
d_7	0	ST	Y	23.399 s	4.1 V	Y	Y	Y	Y
d_8	0	ST	Y	25.985 s	4.1 V	Y	Y	Y	Y
d_9	0	ST	Y	29.450 s	4.1 V	Y	Y	Y	Y
d_{10}	0	ST	Y	31.985 s	4.1 V	Y	Y	Y	Y
d_{11}	0	ST	Y	35.192 s	4.1 V	N	Y	Y	Y
d_{12}	0	ST	Y	5.828 s	3.2 V	Y	Y	Y	Y
d_{13}	0	ST	Y	8.914 s	3.2 V	Y	Y	Y	Y
d_{14}	0	ST	Y	11.328 s	3.2 V	Y	Y	Y	Y
d_{15}	0	ST	Y	14.950 s	3.2 V	Y	Y	Y	Y
d_{16}	0	ST	Y	15.036 s	3.2 V	Y	Y	Y	Y
d_{17}	0	ST	Y	20.243 s	3.2 V	Y	Y	Y	Y
d_{18}	0	ST	Y	23.399 s	3.2 V	N	Y	Y	Y
d_{19}	0	ST	Y	25.985 s	3.2 V	Y	Y	Y	Y
d_{20}	0	ST	Y	29.450 s	3.2 V	Y	Y	Y	Y
d_{21}	0	ST	Y	31.985 s	3.2 V	Y	Y	Y	Y
d_{22}	0	ST	Y	35.192 s	3.2 V	N	Y	Y	Y
d_{23}	0	ST	Y	5.828 s	2.4 V	Y	Y	Y	Y
d_{24}	0	ST	Y	8.914 s	2.4 V	Y	Y	Y	Y
d_{25}	0	ST	Y	11.328 s	2.4 V	Y	Y	Y	Y
d_{26}	0	ST	Y	14.950 s	2.4 V	Y	Y	Y	Y
d_{27}	0	ST	Y	15.036 s	2.4 V	Y	Y	Y	Y
d_{28}	0	ST	Y	20.243 s	2.4 V	N	Y	Y	Y
d_{29}	0	ST	Y	23.399 s	2.4 V	N	Y	Y	Y
d_{30}	0	ST	Y	25.985 s	2.4 V	Y	Y	Y	Y
d_{31}	0	ST	Y	29.450 s	2.4 V	Y	Y	Y	Y
d_{32}	0	ST	Y	31.985 s	2.4 V	Y	Y	Y	Y
d_{33}	0	ST	Y	35.192 s	2.4 V	Y	Y	Y	Y

Table A.1. The Collected Data of Task 1

	A	B	C	D	E	H	J	L	N
$\mathbf{d_1}$	0	ST	Y	11.000 s	4.1 V	Y	Y	Y	Y
$\mathbf{d_2}$	0	ST	Y	14.914 s	4.1 V	Y	Y	Y	Y
$\mathbf{d_3}$	0	ST	Y	17.328 s	4.1 V	Y	Y	Y	Y
$\mathbf{d_4}$	0	ST	Y	19.914 s	4.1 V	Y	Y	Y	Y
$\mathbf{d_5}$	0	ST	Y	23.157 s	4.1 V	Y	Y	Y	Y
$\mathbf{d_6}$	0	ST	Y	26.414 s	4.1 V	Y	Y	Y	Y
$\mathbf{d_7}$	0	ST	Y	29.399 s	4.1 V	N	Y	Y	Y
$\mathbf{d_8}$	0	ST	Y	31.950 s	4.1 V	Y	Y	Y	Y
$\mathbf{d_9}$	0	ST	Y	35.450 s	4.1 V	N	Y	Y	Y
$\mathbf{d_{10}}$	0	ST	Y	37.985 s	4.1 V	N	Y	Y	Y
$\mathbf{d_{11}}$	0	ST	Y	41.192 s	4.1 V	N	Y	Y	Y
$\mathbf{d_{12}}$	0	ST	Y	11.000 s	3.2 V	Y	Y	Y	Y
$\mathbf{d_{13}}$	0	ST	Y	14.914 s	3.2 V	Y	Y	Y	Y
$\mathbf{d_{14}}$	0	ST	Y	17.328 s	3.2 V	Y	Y	Y	Y
$\mathbf{d_{15}}$	0	ST	Y	19.914 s	3.2 V	Y	Y	Y	Y
$\mathbf{d_{16}}$	0	ST	Y	23.157 s	3.2 V	Y	Y	Y	Y
$\mathbf{d_{17}}$	0	ST	Y	26.414 s	3.2 V	N	Y	Y	Y
$\mathbf{d_{18}}$	0	ST	Y	29.399 s	3.2 V	N	Y	Y	Y
$\mathbf{d_{19}}$	0	ST	Y	31.950 s	3.2 V	Y	Y	Y	Y
$\mathbf{d_{20}}$	0	ST	Y	35.450 s	3.2 V	N	Y	Y	Y
$\mathbf{d_{21}}$	0	ST	Y	37.985 s	3.2 V	N	Y	Y	Y
$\mathbf{d_{22}}$	0	ST	Y	41.192 s	3.2 V	N	Y	Y	Y
$\mathbf{d_{23}}$	0	ST	Y	11.000 s	2.5 V	Y	Y	Y	Y
$\mathbf{d_{24}}$	0	ST	Y	14.914 s	2.5 V	Y	Y	Y	Y
$\mathbf{d_{25}}$	0	ST	Y	17.328 s	2.5 V	Y	Y	Y	Y
$\mathbf{d_{26}}$	0	ST	Y	19.914 s	2.5 V	Y	Y	Y	Y
$\mathbf{d_{27}}$	0	ST	Y	23.157 s	2.5 V	Y	Y	Y	Y
$\mathbf{d_{28}}$	0	ST	Y	26.414 s	2.5 V	Y	Y	Y	Y
$\mathbf{d_{29}}$	0	ST	Y	29.399 s	2.5 V	N	Y	Y	Y
$\mathbf{d_{30}}$	0	ST	Y	31.950 s	2.5 V	Y	Y	Y	Y
$\mathbf{d_{31}}$	0	ST	Y	35.450 s	2.5 V	N	Y	Y	Y
$\mathbf{d_{32}}$	0	ST	Y	37.985 s	2.5 V	N	Y	Y	Y
$\mathbf{d_{33}}$	0	ST	Y	41.192 s	2.5 V	N	Y	Y	Y

Table A.2. The Collected Data of Task 2

	A	B	C	D	E	H	J	L	N
$\mathbf{d_1}$	0	ST	Y	6.500 s	4.1 V	Y	Y	Y	Y
$\mathbf{d_2}$	0	ST	Y	10.000 s	4.1 V	Y	Y	Y	Y
$\mathbf{d_3}$	0	ST	Y	14.157 s	4.1 V	Y	Y	Y	Y
$\mathbf{d_4}$	0	ST	Y	17.536 s	4.1 V	Y	Y	Y	Y
$\mathbf{d_5}$	0	ST	Y	19.657 s	4.1 V	Y	Y	Y	Y
$\mathbf{d_6}$	0	ST	Y	22.950 s	4.1 V	Y	Y	Y	Y
$\mathbf{d_7}$	0	ST	Y	26.657 s	4.1 V	Y	Y	Y	Y
$\mathbf{d_8}$	0	ST	Y	29.192 s	4.1 V	Y	Y	Y	Y
$\mathbf{d_9}$	0	ST	Y	6.500 s	3.2 V	Y	Y	Y	Y
$\mathbf{d_{10}}$	0	ST	Y	10.000 s	3.2 V	Y	Y	Y	Y
$\mathbf{d_{11}}$	0	ST	Y	14.157 s	3.2 V	Y	Y	Y	Y
$\mathbf{d_{12}}$	0	ST	Y	17.536 s	3.2 V	Y	Y	Y	Y
$\mathbf{d_{13}}$	0	ST	Y	19.657 s	3.2 V	Y	Y	Y	Y
$\mathbf{d_{14}}$	0	ST	Y	22.950 s	3.2 V	Y	Y	Y	Y
$\mathbf{d_{15}}$	0	ST	Y	26.657 s	3.2 V	N	Y	Y	Y
$\mathbf{d_{16}}$	0	ST	Y	29.192 s	3.2 V	Y	Y	Y	Y
$\mathbf{d_{17}}$	0	ST	Y	6.500 s	2.3 V	Y	Y	Y	Y
$\mathbf{d_{18}}$	0	ST	Y	10.000 s	2.3 V	Y	Y	Y	Y
$\mathbf{d_{19}}$	0	ST	Y	14.157 s	2.3 V	Y	Y	Y	Y
$\mathbf{d_{20}}$	0	ST	Y	17.536 s	2.3 V	Y	Y	Y	Y
$\mathbf{d_{21}}$	0	ST	Y	19.657 s	2.3 V	Y	Y	Y	Y
$\mathbf{d_{22}}$	0	ST	Y	22.950 s	2.3 V	Y	Y	Y	Y
$\mathbf{d_{23}}$	0	ST	Y	26.657 s	2.3 V	Y	Y	Y	Y
$\mathbf{d_{24}}$	0	ST	Y	29.192 s	2.3 V	N	Y	Y	Y

Table A.3. The Collected Data of Task 3

	A	B	C	D	E	H	J	L	N
$\mathbf{d_1}$	0	ST	Y	19.950 s	4.1 V	Y	Y	Y	Y
$\mathbf{d_2}$	0	ST	Y	22.778 s	4.1 V	Y	Y	Y	Y
$\mathbf{d_3}$	0	ST	Y	25.071 s	4.1 V	Y	Y	Y	Y
$\mathbf{d_4}$	0	ST	Y	27.657 s	4.1 V	N	Y	Y	Y
$\mathbf{d_5}$	0	ST	Y	32.107 s	4.1 V	N	Y	Y	Y
$\mathbf{d_6}$	0	ST	Y	34.778 s	4.1 V	N	Y	Y	Y
$\mathbf{d_7}$	0	ST	Y	37.314 s	4.1 V	Y	Y	Y	Y
$\mathbf{d_8}$	0	ST	Y	40.107 s	4.1 V	Y	Y	Y	Y
$\mathbf{d_9}$	0	ST	Y	42.642 s	4.1 V	Y	Y	Y	Y
$\mathbf{d_{10}}$	0	ST	Y	19.950 s	3.1 V	Y	Y	Y	Y
$\mathbf{d_{11}}$	0	ST	Y	22.778 s	3.1 V	Y	Y	Y	Y
$\mathbf{d_{12}}$	0	ST	Y	25.071 s	3.1 V	Y	Y	Y	Y
$\mathbf{d_{13}}$	0	ST	Y	27.657 s	3.1 V	Y	Y	Y	Y
$\mathbf{d_{14}}$	0	ST	Y	32.107 s	3.1 V	N	Y	Y	Y
$\mathbf{d_{15}}$	0	ST	Y	34.778 s	3.1 V	N	Y	Y	Y
$\mathbf{d_{16}}$	0	ST	Y	37.314 s	3.1 V	N	Y	Y	Y
$\mathbf{d_{17}}$	0	ST	Y	40.107 s	3.1 V	N	Y	Y	Y
$\mathbf{d_{18}}$	0	ST	Y	42.642 s	3.1 V	N	Y	Y	Y
$\mathbf{d_{19}}$	0	ST	Y	19.950 s	2.4 V	Y	Y	Y	Y
$\mathbf{d_{20}}$	0	ST	Y	22.778 s	2.4 V	Y	Y	Y	Y
$\mathbf{d_{21}}$	0	ST	Y	25.071 s	2.4 V	Y	Y	Y	Y
$\mathbf{d_{22}}$	0	ST	Y	27.657 s	2.4 V	Y	Y	Y	Y
$\mathbf{d_{23}}$	0	ST	Y	32.107 s	2.4 V	Y	Y	Y	Y
$\mathbf{d_{24}}$	0	ST	Y	34.778 s	2.4 V	N	Y	Y	Y
$\mathbf{d_{25}}$	0	ST	Y	37.314 s	2.4 V	Y	Y	Y	Y
$\mathbf{d_{26}}$	0	ST	Y	40.107 s	2.4 V	Y	Y	Y	Y
$\mathbf{d_{27}}$	0	ST	Y	42.642 s	2.4 V	Y	Y	Y	Y

Table A.4. The Collected Data of Task 4

	A	B	C	D	E	H	J	L	N
$\mathbf{d_1}$	0	ST	Y	5.000 s	4.1 V	Y	Y	Y	Y
$\mathbf{d_2}$	0	ST	Y	7.414 s	4.1 V	Y	Y	Y	Y
$\mathbf{d_3}$	0	ST	Y	11.328 s	4.1 V	Y	Y	Y	Y
$\mathbf{d_4}$	0	ST	Y	13.450 s	4.1 V	Y	Y	Y	Y
$\mathbf{d_5}$	0	ST	Y	17.899 s	4.1 V	Y	Y	Y	Y
$\mathbf{d_6}$	0	ST	Y	20.243 s	4.1 V	Y	Y	Y	Y
$\mathbf{d_7}$	0	ST	Y	23.571 s	4.1 V	Y	Y	Y	Y
$\mathbf{d_8}$	0	ST	Y	25.414 s	4.1 V	Y	Y	Y	Y
$\mathbf{d_9}$	0	ST	Y	28.399 s	4.1 V	N	Y	Y	Y
$\mathbf{d_{10}}$	0	ST	Y	32.385 s	4.1 V	Y	Y	Y	Y
$\mathbf{d_{11}}$	0	ST	Y	5.000 s	3.2 V	Y	Y	Y	Y
$\mathbf{d_{12}}$	0	ST	Y	7.414 s	3.2 V	Y	Y	Y	Y
$\mathbf{d_{13}}$	0	ST	Y	11.328 s	3.2 V	Y	Y	Y	Y
$\mathbf{d_{14}}$	0	ST	Y	13.450 s	3.2 V	Y	Y	Y	Y
$\mathbf{d_{15}}$	0	ST	Y	17.899 s	3.2 V	Y	Y	Y	Y
$\mathbf{d_{16}}$	0	ST	Y	20.243 s	3.2 V	Y	Y	Y	Y
$\mathbf{d_{17}}$	0	ST	Y	23.571 s	3.2 V	Y	Y	Y	Y
$\mathbf{d_{18}}$	0	ST	Y	25.414 s	3.2 V	Y	Y	Y	Y
$\mathbf{d_{19}}$	0	ST	Y	28.399 s	3.2 V	Y	Y	Y	Y
$\mathbf{d_{20}}$	0	ST	Y	32.385 s	3.2 V	N	Y	Y	Y
$\mathbf{d_{21}}$	0	ST	Y	5.000 s	2.3 V	Y	Y	Y	Y
$\mathbf{d_{22}}$	0	ST	Y	7.414 s	2.3 V	Y	Y	Y	Y
$\mathbf{d_{23}}$	0	ST	Y	11.328 s	2.3 V	Y	Y	Y	Y
$\mathbf{d_{24}}$	0	ST	Y	13.450 s	2.3 V	Y	Y	Y	Y
$\mathbf{d_{25}}$	0	ST	Y	17.899 s	2.3 V	Y	Y	Y	Y
$\mathbf{d_{26}}$	0	ST	Y	20.243 s	2.3 V	Y	Y	Y	Y
$\mathbf{d_{27}}$	0	ST	Y	23.571 s	2.3 V	Y	Y	Y	Y
$\mathbf{d_{28}}$	0	ST	Y	25.414 s	2.3 V	Y	Y	Y	Y
$\mathbf{d_{29}}$	0	ST	Y	28.399 s	2.3 V	N	Y	Y	Y
$\mathbf{d_{30}}$	0	ST	Y	32.385 s	2.3 V	Y	Y	Y	Y

Table A.5. The Collected Data of Task 5

	A	B	C	D	E	H	J	L	N
$\mathbf{d_1}$	0	ST	Y	16.571 s	4.1 V	Y	Y	Y	Y
$\mathbf{d_2}$	0	ST	Y	18.899 s	4.1 V	Y	Y	Y	Y
$\mathbf{d_3}$	0	ST	Y	22.071 s	4.1 V	Y	Y	Y	Y
$\mathbf{d_4}$	0	ST	Y	26.642 s	4.1 V	Y	Y	Y	Y
$\mathbf{d_5}$	0	ST	Y	28.521 s	4.1 V	Y	Y	Y	Y
$\mathbf{d_6}$	0	ST	Y	32.314 s	4.1 V	N	Y	Y	Y
$\mathbf{d_7}$	0	ST	Y	35.471 s	4.1 V	Y	Y	Y	Y
$\mathbf{d_8}$	0	ST	Y	37.107 s	4.1 V	Y	Y	Y	Y
$\mathbf{d_9}$	0	ST	Y	40.485 s	4.1 V	N	Y	Y	Y
$\mathbf{d_{10}}$	0	ST	Y	42.092 s	4.1 V	N	Y	Y	Y
$\mathbf{d_{11}}$	0	ST	Y	16.571 s	3.2 V	Y	Y	Y	Y
$\mathbf{d_{12}}$	0	ST	Y	18.899 s	3.2 V	Y	Y	Y	Y
$\mathbf{d_{13}}$	0	ST	Y	22.071 s	3.2 V	Y	Y	Y	Y
$\mathbf{d_{14}}$	0	ST	Y	26.642 s	3.2 V	Y	Y	Y	Y
$\mathbf{d_{15}}$	0	ST	Y	28.521 s	3.2 V	Y	Y	Y	Y
$\mathbf{d_{16}}$	0	ST	Y	32.314 s	3.2 V	N	Y	Y	Y
$\mathbf{d_{17}}$	0	ST	Y	35.471 s	3.2 V	Y	Y	Y	Y
$\mathbf{d_{18}}$	0	ST	Y	37.107 s	3.2 V	Y	Y	Y	Y
$\mathbf{d_{19}}$	0	ST	Y	40.485 s	3.2 V	Y	Y	Y	Y
$\mathbf{d_{20}}$	0	ST	Y	42.092 s	3.2 V	N	Y	Y	Y
$\mathbf{d_{21}}$	0	ST	Y	16.571 s	2.5 V	Y	Y	Y	Y
$\mathbf{d_{22}}$	0	ST	Y	18.899 s	2.5 V	Y	Y	Y	Y
$\mathbf{d_{23}}$	0	ST	Y	22.071 s	2.5 V	Y	Y	Y	Y
$\mathbf{d_{24}}$	0	ST	Y	26.642 s	2.5 V	Y	Y	Y	Y
$\mathbf{d_{25}}$	0	ST	Y	28.521 s	2.5 V	Y	Y	Y	Y
$\mathbf{d_{26}}$	0	ST	Y	32.314 s	2.5 V	N	Y	Y	Y
$\mathbf{d_{27}}$	0	ST	Y	35.471 s	2.5 V	Y	Y	Y	Y
$\mathbf{d_{28}}$	0	ST	Y	37.107 s	2.5 V	N	Y	Y	Y
$\mathbf{d_{29}}$	0	ST	Y	40.485 s	2.5 V	N	Y	Y	Y
$\mathbf{d_{30}}$	0	ST	Y	42.092 s	2.5 V	Y	Y	Y	Y

Table A.6. The Collected Data of Task 6

	A	B	C	D	E	H	J	L
$\mathbf{d_1}$	0	ST	Y	1.207 s	4.1 V	Y	Y	Y
$\mathbf{d_2}$	0	ST	Y	5.707 s	4.1 V	Y	Y	Y
$\mathbf{d_3}$	0	ST	Y	8.707 s	4.1 V	Y	Y	Y
$\mathbf{d_4}$	0	ST	Y	11.621 s	4.1 V	Y	Y	Y
$\mathbf{d_5}$	0	ST	Y	13.121 s	4.1 V	Y	Y	Y
$\mathbf{d_6}$	0	ST	Y	16.207 s	4.1 V	Y	Y	Y
$\mathbf{d_7}$	0	ST	Y	20.035 s	4.1 V	Y	Y	Y
$\mathbf{d_8}$	0	ST	Y	22.899 s	4.1 V	Y	Y	Y
$\mathbf{d_9}$	0	ST	Y	25.692 s	4.1 V	Y	Y	Y
$\mathbf{d_{10}}$	0	ST	Y	29.399 s	4.1 V	Y	Y	Y
$\mathbf{d_{11}}$	0	ST	Y	32.228 s	4.1 V	N	Y	Y
$\mathbf{d_{12}}$	0	ST	Y	35.730 s	4.1 V	N	Y	Y
$\mathbf{d_{13}}$	0	ST	Y	36.899 s	4.1 V	Y	Y	Y
$\mathbf{d_{14}}$	0	ST	Y	39.107 s	4.1 V	Y	Y	Y
$\mathbf{d_{15}}$	0	ST	Y	1.207 s	3.2 V	Y	Y	Y
$\mathbf{d_{16}}$	0	ST	Y	5.707 s	3.2 V	Y	Y	Y
$\mathbf{d_{17}}$	0	ST	Y	8.707 s	3.2 V	Y	Y	Y
$\mathbf{d_{18}}$	0	ST	Y	11.621 s	3.2 V	Y	Y	Y
$\mathbf{d_{19}}$	0	ST	Y	13.121 s	3.2 V	Y	Y	Y
$\mathbf{d_{20}}$	0	ST	Y	16.207 s	3.2 V	Y	Y	Y
$\mathbf{d_{21}}$	0	ST	Y	20.035 s	3.2 V	Y	Y	Y
$\mathbf{d_{22}}$	0	ST	Y	22.899 s	3.2 V	Y	Y	Y
$\mathbf{d_{23}}$	0	ST	Y	25.692 s	3.2 V	Y	Y	Y
$\mathbf{d_{24}}$	0	ST	Y	29.399 s	3.2 V	Y	Y	Y
$\mathbf{d_{25}}$	0	ST	Y	32.228 s	3.2 V	N	Y	Y
$\mathbf{d_{26}}$	0	ST	Y	35.730 s	3.2 V	Y	Y	Y
$\mathbf{d_{27}}$	0	ST	Y	36.899 s	3.2 V	Y	Y	Y
$\mathbf{d_{28}}$	0	ST	Y	39.107 s	3.2 V	N	Y	Y
$\mathbf{d_{29}}$	0	ST	Y	1.207 s	2.4 V	Y	Y	Y
$\mathbf{d_{30}}$	0	ST	Y	5.707 s	2.4 V	Y	Y	Y
$\mathbf{d_{31}}$	0	ST	Y	8.707 s	2.4 V	Y	Y	Y
$\mathbf{d_{32}}$	0	ST	Y	11.621 s	2.4 V	Y	Y	Y
$\mathbf{d_{33}}$	0	ST	Y	13.121 s	2.4 V	Y	Y	Y
$\mathbf{d_{34}}$	0	ST	Y	16.207 s	2.4 V	Y	Y	Y
$\mathbf{d_{35}}$	0	ST	Y	20.035 s	2.4 V	Y	Y	Y
$\mathbf{d_{36}}$	0	ST	Y	22.899 s	2.4 V	Y	Y	Y
$\mathbf{d_{37}}$	0	ST	Y	25.692 s	2.4 V	Y	Y	Y
$\mathbf{d_{38}}$	0	ST	Y	29.399 s	2.4 V	Y	Y	Y
$\mathbf{d_{39}}$	0	ST	Y	32.228 s	2.4 V	N	Y	Y
$\mathbf{d_{40}}$	0	ST	Y	35.730 s	2.4 V	Y	Y	Y
$\mathbf{d_{41}}$	0	ST	Y	36.899 s	2.4 V	Y	Y	Y
$\mathbf{d_{42}}$	0	ST	Y	39.107 s	2.4 V	Y	Y	Y

Table A.7. The Collected Data of Task 7

Appendix B

Partial Collected Data with Assumptions and Facts

	A	B	C	D	E	H	J	L	N
$\mathbf{d_{1-1}}$	0	ST	Y	5.828 s	4.1 V	Y	Y	Y	Y
$\mathbf{d_{1-2}}$	0	ST	Y	2.000 s	4.1 V	Y	Y	Y	Y
$\mathbf{d_{1-3}}$	0	ST	Y	5.828 s	3.2 V	Y	Y	Y	Y
$\mathbf{d_{1-4}}$	0	ST	Y	2.000 s	3.2 V	Y	Y	Y	Y
$\mathbf{d_{1-5}}$	0	ST	Y	5.828 s	2.4 V	Y	Y	Y	Y
$\mathbf{d_{1-6}}$	0	ST	Y	2.000 s	2.4 V	Y	Y	Y	Y
$\mathbf{d_{2-1}}$	0	ST	Y	8.914 s	4.1 V	Y	Y	Y	Y
$\mathbf{d_{2-2}}$	0	ST	Y	5.000 s	4.1 V	Y	Y	Y	Y
$\mathbf{d_{2-3}}$	0	ST	Y	2.000 s	4.1 V	Y	Y	Y	Y
$\mathbf{d_{2-4}}$	0	ST	Y	8.914 s	3.2 V	Y	Y	Y	Y
$\mathbf{d_{2-5}}$	0	ST	Y	5.000 s	3.2 V	Y	Y	Y	Y
$\mathbf{d_{2-6}}$	0	ST	Y	2.000 s	3.2 V	Y	Y	Y	Y
$\mathbf{d_{2-7}}$	0	ST	Y	8.914 s	2.4 V	Y	Y	Y	Y
$\mathbf{d_{2-8}}$	0	ST	Y	5.000 s	2.4 V	Y	Y	Y	Y
$\mathbf{d_{2-9}}$	0	ST	Y	2.000 s	2.4 V	Y	Y	Y	Y
.....									
$\mathbf{d_{11-1}}$	0	ST	Y	35.192 s	4.1 V	N	Y	Y	Y
$\mathbf{d_{11-2}}$	0	ST	Y	37.000 s	4.1 V	N	Y	Y	Y
$\mathbf{d_{11-3}}$	0	ST	Y	40.000 s	4.1 V	N	Y	Y	Y
$\mathbf{d_{11-4}}$	0	ST	Y	43.000 s	4.1 V	N	Y	Y	Y
$\mathbf{d_{11-5}}$	0	ST	Y	35.192 s	3.2 V	N	Y	Y	Y
$\mathbf{d_{11-6}}$	0	ST	Y	37.000 s	3.2 V	N	Y	Y	Y
$\mathbf{d_{11-7}}$	0	ST	Y	40.000 s	3.2 V	N	Y	Y	Y
$\mathbf{d_{11-8}}$	0	ST	Y	43.000 s	3.2 V	N	Y	Y	Y
$\mathbf{d_{11-9}}$	0	ST	Y	35.192 s	2.4 V	N	Y	Y	Y
$\mathbf{d_{11-10}}$	0	ST	Y	37.000 s	2.4 V	N	Y	Y	Y
$\mathbf{d_{11-11}}$	0	ST	Y	40.000 s	2.4 V	N	Y	Y	Y
$\mathbf{d_{11-12}}$	0	ST	Y	43.000 s	2.4 V	N	Y	Y	Y
.....									

Table B.1. The Partial Collected Data of Task 1 with Assumption 1 and 2

	A	B	C	D	E	H	J	L	N
$\mathbf{d_1}$	0	ST	Y	2.000 s	4.1 V	N	Y	Y	N
$\mathbf{d_2}$	0	ST	Y	2.000 s	4.1 V	N	Y	N	Y
$\mathbf{d_3}$	0	ST	Y	2.000 s	4.1 V	N	Y	N	N
$\mathbf{d_4}$	0	ST	Y	2.000 s	4.1 V	N	N	Y	Y
$\mathbf{d_5}$	0	ST	Y	2.000 s	4.1 V	N	N	Y	N
$\mathbf{d_6}$	0	ST	Y	2.000 s	4.1 V	N	N	N	Y
$\mathbf{d_7}$	0	ST	Y	2.000 s	4.1 V	N	N	N	N
$\mathbf{d_8}$	0	ST	Y	5.000 s	4.1 V	N	Y	Y	N
$\mathbf{d_9}$	0	ST	Y	5.000 s	4.1 V	N	Y	N	Y
$\mathbf{d_{10}}$	0	ST	Y	5.000 s	4.1 V	N	Y	N	N
$\mathbf{d_{11}}$	0	ST	Y	5.000 s	4.1 V	N	N	Y	Y
$\mathbf{d_{12}}$	0	ST	Y	5.000 s	4.1 V	N	N	Y	N
$\mathbf{d_{13}}$	0	ST	Y	5.000 s	4.1 V	N	N	N	Y
$\mathbf{d_{14}}$	0	ST	Y	5.000 s	4.1 V	N	N	N	N
.....									
$\mathbf{d_{106}}$	0	ST	Y	2.000 s	3.2 V	N	Y	Y	N
$\mathbf{d_{107}}$	0	ST	Y	2.000 s	3.2 V	N	Y	N	Y
$\mathbf{d_{108}}$	0	ST	Y	2.000 s	3.2 V	N	Y	N	N
$\mathbf{d_{109}}$	0	ST	Y	2.000 s	3.2 V	N	N	Y	Y
$\mathbf{d_{110}}$	0	ST	Y	2.000 s	3.2 V	N	N	Y	N
$\mathbf{d_{111}}$	0	ST	Y	2.000 s	3.2 V	N	N	N	Y
$\mathbf{d_{112}}$	0	ST	Y	2.000 s	3.2 V	N	N	N	N
$\mathbf{d_{113}}$	0	ST	Y	5.000 s	3.2 V	N	Y	Y	N
$\mathbf{d_{114}}$	0	ST	Y	5.000 s	3.2 V	N	Y	N	Y
$\mathbf{d_{115}}$	0	ST	Y	5.000 s	3.2 V	N	Y	N	N
$\mathbf{d_{116}}$	0	ST	Y	5.000 s	3.2 V	N	N	Y	Y
$\mathbf{d_{117}}$	0	ST	Y	5.000 s	3.2 V	N	N	Y	N
$\mathbf{d_{118}}$	0	ST	Y	5.000 s	3.2 V	N	N	N	Y
$\mathbf{d_{119}}$	0	ST	Y	5.000 s	3.2 V	N	N	N	N
.....									

Table B.2. The Partial Collected Data of Task 1 with Fact 1

	A	B	C	E	H	J	L	N
$\mathbf{d_1}$	0	ST	N	4.1 V	N	Y	Y	Y
$\mathbf{d_2}$	0	ST	N	4.1 V	N	Y	Y	N
$\mathbf{d_3}$	0	ST	N	4.1 V	N	Y	N	Y
$\mathbf{d_4}$	0	ST	N	4.1 V	N	Y	N	N
$\mathbf{d_5}$	0	ST	N	4.1 V	N	N	Y	Y
$\mathbf{d_6}$	0	ST	N	4.1 V	N	N	Y	N
$\mathbf{d_7}$	0	ST	N	4.1 V	N	N	N	Y
$\mathbf{d_8}$	0	ST	N	4.1 V	N	N	N	N
$\mathbf{d_9}$	0	ST	N	3.2 V	N	Y	Y	Y
$\mathbf{d_{10}}$	0	ST	N	3.2 V	N	Y	Y	N
$\mathbf{d_{11}}$	0	ST	N	3.2 V	N	Y	N	Y
$\mathbf{d_{12}}$	0	ST	N	3.2 V	N	Y	N	N
$\mathbf{d_{13}}$	0	ST	N	3.2 V	N	N	Y	Y
$\mathbf{d_{14}}$	0	ST	N	3.2 V	N	N	Y	N
$\mathbf{d_{15}}$	0	ST	N	3.2 V	N	N	N	Y
$\mathbf{d_{16}}$	0	ST	N	3.2 V	N	N	N	N
$\mathbf{d_{17}}$	0	ST	N	2.4 V	N	Y	Y	Y
$\mathbf{d_{18}}$	0	ST	N	2.4 V	N	Y	Y	N
$\mathbf{d_{19}}$	0	ST	N	2.4 V	N	Y	N	Y
$\mathbf{d_{20}}$	0	ST	N	2.4 V	N	Y	N	N
$\mathbf{d_{21}}$	0	ST	N	2.4 V	N	N	Y	Y
$\mathbf{d_{22}}$	0	ST	N	2.4 V	N	N	Y	N
$\mathbf{d_{23}}$	0	ST	N	2.4 V	N	N	N	Y
$\mathbf{d_{24}}$	0	ST	N	2.4 V	N	N	N	N

Table B.3. The Partial Collected Data of Task 1 with Fact 2

Appendix C

The Updated CPTs in the First Iteration

P(F=Yes|DE)		E		
		2∼3 V	3∼4 V	4∼5 V
D	0∼3 sec	0.8117722755	0.8747023052	0.9962777837
	3∼6 sec	0.7736172227	0.8117722755	0.8747023052
	6∼9 sec	0.7283683977	0.7561541939	0.7972199011
	9∼12 sec	0.6838074336	0.7056836980	0.7357424361
	12∼15 sec	0.6367561849	0.6550119368	0.6788153447
	15∼18 sec	0.5844126290	0.6003694027	0.6203663391
	18∼21 sec	0.5033912400	0.5197130785	0.5394432575
	21∼24 sec	0.4529612221	0.4686487586	0.4871106567
	24∼27 sec	0.4200999745	0.4341247229	0.4503004266
	27∼30 sec	0.3397708262	0.3533369678	0.3687625626
	30∼33 sec	0.2404151423	0.2537649753	0.2687968292
	33∼36 sec	0.1118519554	0.1258304413	0.1415272305
	36∼39 sec	0.0567033099	0.0696432707	0.0841050111
	39∼42 sec	0.0450338792	0.0567033099	0.0696432707
	Larger than 42 sec	0.0344364765	0.0450338792	0.0567033099

Table C.1. The Updated CPT of Node F in Task 1

P(F=Yes|DE)		E		
		2∼3 V	3∼4 V	4∼5 V
D	0∼3 sec	0.7738836610	0.8494300468	0.9955256768
	3∼6 sec	0.7281504180	0.7738836610	0.8494300468
	6∼9 sec	0.6972462743	0.7281504180	0.7738836610
	9∼12 sec	0.6747648323	0.6972462743	0.7281504180
	12∼15 sec	0.6253965341	0.6441958693	0.6687231443
	15∼18 sec	0.5700691226	0.5865401584	0.6071988534
	18∼21 sec	0.5052894685	0.5202975300	0.5385530416
	21∼24 sec	0.4263129250	0.4404657219	0.4572731463
	24∼27 sec	0.3183037869	0.3328705582	0.3498387824
	27∼30 sec	0.2197709186	0.2353820995	0.2532826674
	30∼33 sec	0.2059486965	0.2197709186	0.2353820995
	33∼36 sec	0.0710159115	0.0857713137	0.1024053865
	36∼39 sec	0.0605974279	0.0744624334	0.0899551823
	39∼42 sec	0.0498104946	0.0627754939	0.0771578973
	Larger than 42 sec	0.0380481875	0.0498104946	0.0627754939

Table C.2. The Updated CPT of Node F in Task 2

P(F=Yes|DE)		E		
		2∼3 V	3∼4 V	4∼5 V
D	0∼3 sec	0.7861905468	0.8576389191	0.9957699743
	3∼6 sec	0.7429188327	0.7861905468	0.8576389191
	6∼9 sec	0.7136592189	0.7429188327	0.7861905468
	9∼12 sec	0.6651829108	0.6883056245	0.7201055957
	12∼15 sec	0.6132928013	0.6326713258	0.6579698828
	15∼18 sec	0.5546837151	0.5717063684	0.5930748917
	18∼21 sec	0.4854430678	0.5010020212	0.5199493406
	21∼24 sec	0.4001155686	0.4148439521	0.4323625798
	24∼27 sec	0.2840022397	0.2992071397	0.3169540187
	27∼30 sec	0.1866872415	0.2018335491	0.2192607740
	30∼33 sec	0.0754241119	0.0900239648	0.1066931709
	33∼36 sec	0.0624922551	0.0754241119	0.0900239648
	36∼39 sec	0.0509249413	0.0624922551	0.0754241119
	39∼42 sec	0.0404883904	0.0509249413	0.0624922551
	Larger than 42 sec	0.0309995258	0.0404883904	0.0509249413

Table C.3. The Updated CPT of Node F in Task 3

P(F=Yes|DE)		E		
		2∼3 V	3∼4 V	4∼5 V
D	0∼3 sec	0.7600636336	0.8402118870	0.9952513427
	3∼6 sec	0.7115662147	0.7600636336	0.8402118870
	6∼9 sec	0.6788153447	0.7115662147	0.7600636336
	9∼12 sec	0.6550119368	0.6788153447	0.7115662147
	12∼15 sec	0.6367561849	0.6550119368	0.6788153447
	15∼18 sec	0.6221525106	0.6367561849	0.6550119368
	18∼21 sec	0.6100524361	0.6221525106	0.6367561849
	21∼24 sec	0.5600199740	0.5712350708	0.5844126290
	24∼27 sec	0.5006659463	0.5113555994	0.5236399094
	27∼30 sec	0.4164530934	0.4275467579	0.4400812278
	30∼33 sec	0.3360793331	0.3477772476	0.3608183913
	33∼36 sec	0.2751719917	0.2872805985	0.3006182667
	36∼39 sec	0.2572874950	0.2687807862	0.2812045257
	39∼42 sec	0.1852561426	0.1966661008	0.2088903726
	Larger than 42 sec	0.1082037100	0.1193625242	0.1312962945

Table C.4. The Updated CPT of Node F in Task 4

P(F=Yes|DE)		E		
		2∼3 V	3∼4 V	4∼5 V
D	0∼3 sec	0.8117722755	0.8747023052	0.9962777837
	3∼6 sec	0.7736172227	0.8117722755	0.8747023052
	6∼9 sec	0.7283683977	0.7561541939	0.7972199011
	9∼12 sec	0.6838074336	0.7056836980	0.7357424361
	12∼15 sec	0.6367561849	0.6550119368	0.6788153447
	15∼18 sec	0.5844126290	0.6003694027	0.6203663391
	18∼21 sec	0.5236399094	0.5381386034	0.5557544537
	21∼24 sec	0.4502838771	0.4639100042	0.4800666675
	24∼27 sec	0.3581674864	0.3713665217	0.3867265554
	27∼30 sec	0.2275959320	0.2420977364	0.2587403930
	30∼33 sec	0.1705895868	0.1844684384	0.2001932332
	33∼36 sec	0.0645858985	0.0779656679	0.0930651809
	36∼39 sec	0.0526167070	0.0645858985	0.0779656679
	39∼42 sec	0.0418191989	0.0526167070	0.0645858985
	Larger than 42 sec	0.0320057813	0.0418191989	0.0526167070

Table C.5. The Updated CPT of Node F in Task 5

P(F=Yes|DE)		E		
		2∼3 V	3∼4 V	4∼5 V
D	0∼3 sec	0.7861905468	0.8576389191	0.9957699743
	3∼6 sec	0.7429188327	0.7861905468	0.8576389191
	6∼9 sec	0.7136592189	0.7429188327	0.7861905468
	9∼12 sec	0.6923550020	0.7136592189	0.7429188327
	12∼15 sec	0.6759760012	0.6923550020	0.7136592189
	15∼18 sec	0.6628307771	0.6759760012	0.6923550020
	18∼21 sec	0.6214481238	0.6332318794	0.6474383217
	21∼24 sec	0.5741163504	0.5850217333	0.5978166044
	24∼27 sec	0.5183776193	0.5287553718	0.5406573884
	27∼30 sec	0.4507910163	0.4609109412	0.4723006756
	30∼33 sec	0.3360793331	0.3477772476	0.3608183913
	33∼36 sec	0.3253599751	0.3360793331	0.3477772476
	36∼39 sec	0.2172718169	0.2283618941	0.2404151423
	39∼42 sec	0.1267569861	0.1380475008	0.1503143072
	Larger than 42 sec	0.0747565068	0.0857558216	0.0976769246

Table C.6. The Updated CPT of Node F in Task 6

P(F=Yes|DE)		E		
		2∼3 V	3∼4 V	4∼5 V
D	0∼3 sec	0.8401694577	0.8935543450	0.9968364806
	3∼6 sec	0.7962863173	0.8305155596	0.8871098335
	6∼9 sec	0.7586500871	0.7832101791	0.8196097151
	9∼12 sec	0.7230996500	0.7421167532	0.7683207088
	12∼15 sec	0.6871801999	0.7027577059	0.7231198431
	15∼18 sec	0.6490602788	0.6623976466	0.6791406725
	18∼21 sec	0.6070378672	0.6188814705	0.6332757358
	21∼24 sec	0.5592230084	0.5700726795	0.5829114198
	24∼27 sec	0.5032315472	0.5134439822	0.5252611076
	27∼30 sec	0.4357729871	0.4456150809	0.4567939470
	30∼33 sec	0.3262930483	0.3378858438	0.3508784279
	33∼36 sec	0.3079958353	0.3190364013	0.3311722121
	36∼39 sec	0.2420051366	0.2527070142	0.2643277971
	39∼42 sec	0.1257874352	0.1366920776	0.1485801350
	Larger than 42 sec	0.0326685918	0.0428472821	0.0541074355

Table C.7. The Updated CPT of Node F in Task 7

C	K	M	P(G=Yes|CKM)
Yes	Good	Good	0.9856118613
Yes	Good	Bad	0.0007634422
Yes	Bad	Good	0.0011826465
Yes	Bad	Bad	0.0000363324
No	Good	Good	0.500000
No	Good	Bad	0.500000
No	Bad	Good	0.500000
No	Bad	Bad	0.500000

Table C.8. The Updated CPT of Node G (Two Required Skills)

C	K	M	O	P(G=Yes|CKMO)
Yes	Good	Good	Good	0.9765505897
Yes	Good	Good	Bad	0.0010485538
Yes	Good	Bad	Good	0.0010485538
Yes	Good	Bad	Bad	0.0000361309
Yes	Bad	Good	Good	0.0015829227
Yes	Bad	Good	Bad	0.0000423596
Yes	Bad	Bad	Good	0.0000423596
Yes	Bad	Bad	Bad	0.0000029806
No	Good	Good	Good	0.500000
No	Good	Good	Bad	0.500000
No	Good	Bad	Good	0.500000
No	Good	Bad	Bad	0.500000
No	Bad	Good	Good	0.500000
No	Bad	Good	Bad	0.500000
No	Bad	Bad	Good	0.500000
No	Bad	Bad	Bad	0.500000

Table C.9. The Updated CPT of Node G (Three Required Skills)

C	F	G	P(H=Yes|CFG)
Yes	Yes	Yes	0.9890817593
Yes	Yes	No	0.0314478368
Yes	No	Yes	0.2153004347
Yes	No	No	0.0002854063
No	Yes	Yes	0.000000000000001
No	Yes	No	0.000000000000001
No	No	Yes	0.000000000000001
No	No	No	0.000000000000001

Table C.10. The Updated CPT of Node H

References

[1] Ziaul Haque Munim. Autonomous ships: a review, innovative applications and future maritime business models. In *Supply Chain Forum: An International Journal*, volume 20, pages 266–279. Taylor & Francis, 2019.

[2] Kaushik Rajashekara, Qingchun Wang, and Kouki Matsuse. Flying cars: Challenges and propulsion strategies. *IEEE Electrification Magazine*, 4(1):46–57, 2016.

[3] Hyung-Woo Lee, Ki-Chan Kim, and Ju Lee. Review of maglev train technologies. *IEEE transactions on magnetics*, 42(7):1917–1925, 2006.

[4] Jingchao Jiang, Stephen T Newman, and Ray Y Zhong. A review of multiple degrees of freedom for additive manufacturing machines. *International Journal of Computer Integrated Manufacturing*, 34(2):195–211, 2021.

[5] Mikuláš Hajduk, Peter Jenčík, Jaromír Jeznỳ, and Ladislav Vargovčík. Trends in industrial robotics development. In *Applied Mechanics and Materials*, volume 282, pages 1–6. Trans Tech Publ, 2013.

[6] Jun Ni, Jibin Hu, and Changle Xiang. A review for design and dynamics control of unmanned ground vehicle. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, 235(4):1084–1100, 2021.

[7] Jesús Sánchez-García, José M García-Campos, Mario Arzamendia, D Gutierrez Reina, SL Toral, and D Gregor. A survey on unmanned aerial and aquatic vehicle multi-hop networks: Wireless communications, evaluation tools and applications. *Computer Communications*, 119:43–65, 2018.

[8] Di Lu, Chengke Xiong, Hexiong Zhou, Chenxin Lyu, Rui Hu, Caoyang Yu, Zheng Zeng, and Lian Lian. Design, fabrication, and characterization of a multimodal hybrid aerial underwater vehicle. *Ocean Engineering*, 219:108324, 2021.

[9] Nathan Barba, Tom Komarek, Ryan Woolley, Lou Giersch, Vlada Stamenković, Mike Gallagher, and Charles D Edwards. Mars small spacecraft studies: Overview. In *2019 IEEE Aerospace Conference*, pages 1–10. IEEE, 2019.

[10] Anna Frohnwieser, John C Murray, Thomas W Pike, and Anna Wilkinson. Using robots to understand animal cognition. *Journal of the experimental analysis of behavior*, 105(1):14–22, 2016.

[11] Jaishankar Bharatharaj, Loulin Huang, and Ahmed Al-Jumaily. Bio-inspired therapeutic pet robots: Review and future direction. In *2015 10th international conference on information, communications and signal processing (icics)*, pages 1–5. IEEE, 2015.

[12] Cady M Stanton, Peter H Kahn Jr, Rachel L Severson, Jolina H Ruckert, and Brian T Gill. Robotic animals might aid in the social development of children with autism. In *Proceedings of the 3rd ACM/IEEE international conference on Human robot interaction*, pages 271–278, 2008.

[13] Kenji Kaneko, Kensuke Harada, Fumio Kanehiro, Go Miyamori, and Kazuhiko Akachi. Humanoid robot hrp-3. In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2471–2478. IEEE, 2008.

[14] Kerstin Dautenhahn, Chrystopher L Nehaniv, Michael L Walters, Ben Robins, Hatice Kose-Bagci, N Assif, Mike Blow, et al. Kaspar–a minimally expressive humanoid robot for human–robot interaction research. *Applied Bionics and Biomechanics*, 6(3-4):369–397, 2009.

[15] Satoshi Shigemi, Ambarish Goswami, and Prahlad Vadakkepat. Asimo and humanoid robot research at honda. *Humanoid robotics: A reference*, pages 55–90, 2018.

[16] Yunbo Hong, Rongchuan Sun, Rui Lin, Shumei Yu, and Lining Sun. Mopping module design and experiments of a multifunction floor cleaning robot. In *Proceeding of the 11th World Congress on Intelligent Control and Automation*, pages 5097–5102. IEEE, 2014.

[17] Anshu Prakash Murdan and Pawan Kumar Ramkissoon. A smart autonomous floor cleaner with an android-based controller. In *2020 3rd International Conference on Emerging Trends in Electrical, Electronic and Communications Engineering (ELECOM)*, pages 235–239. IEEE, 2020.

[18] Patrick Boissy, Hélène Corriveau, François Michaud, Daniel Labonté, and Marie-Pier Royer. A qualitative study of in-home robotic telepresence for home care of community-living elderly subjects. *Journal of telemedicine and telecare*, 13(2):79–84, 2007.

[19] Cynthia L Breazeal, Anastasia K Ostrowski, Nikhita Singh, and Hae Won Park. Designing social robots for older adults. *Natl. Acad. Eng. Bridge*, 49:22–31, 2019.

[20] Henrique Teixeira, Tiago Silva, Miguel Abreu, and Luís Paulo Reis. Humanoid robot kick in motion ability for playing robotic soccer. In *2020 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, pages 34–39. IEEE, 2020.

[21] George Michalos, Sotiris Makris, and George Chryssolouris. The new assembly system paradigm. *International Journal of Computer Integrated Manufacturing*, 28(12):1252–1261, 2015.

[22] L Chaimowicz, A Cowley, D Gomez-Ibanez, B Grocholsky, MA Hsieh, H Hsu, JF Keller, V Kumar, R Swaminathan, and CJ Taylor. Deploying air-ground multi-robot teams in urban environments. In *Multi-Robot Systems. From Swarms to Intelligent Automata Volume III: Proceedings from the 2005 International Workshop on Multi-Robot Systems*, pages 223–234. Springer, 2005.

[23] Wei Li, Tianguang Zhang, and Kolja Kühnlenz. A vision-guided autonomous quadrotor in an air-ground multi-robot system. In *2011 IEEE International Conference on Robotics and Automation*, pages 2980–2985. IEEE, 2011.

[24] Steven L Waslander. Unmanned aerial and ground vehicle teams: Recent work and open problems. *Autonomous Control Systems and Vehicles: Intelligent Unmanned Systems*, pages 21–36, 2013.

[25] Sondre Engebråten, Kyrre Glette, and Oleg Yakimenko. Field-testing of high-level decentralized controllers for a multi-function drone swarm. In *2018 IEEE 14th International Conference on Control and Automation (ICCA)*, pages 379–386. IEEE, 2018.

[26] Lynne E Parker, Daniela Rus, and Gaurav S Sukhatme. Multiple mobile robot systems. *Springer handbook of robotics*, pages 1335–1384, 2016.

[27] Rachael N Darmanin and Marvin K Bugeja. A review on multi-robot systems categorised by application domain. In *2017 25th mediterranean conference on control and automation (MED)*, pages 701–706. IEEE, 2017.

[28] Michael Rubenstein, Christian Ahler, Nick Hoff, Adrian Cabrera, and Radhika Nagpal. Kilobot: A low cost robot with scalable operations designed for collective behaviors. *Robotics and Autonomous Systems*, 62(7):966–975, 2014.

[29] James McLurkin, Adam McMullen, Nick Robbins, Golnaz Habibi, Aaron Becker, Alvin Chou, Hao Li, Meagan John, Nnena Okeke, Joshua Rykowski, et al. A robot system design for low-cost multi-robot manipulation. In *2014 IEEE/RSJ international conference on intelligent robots and systems*, pages 912–918. IEEE, 2014.

[30] Lynne E Parker. Current research in multirobot systems. *Artificial Life and Robotics*, 7:1–5, 2003.

[31] Avinash Gautam and Sudeept Mohan. A review of research in multi-robot systems. In *2012 IEEE 7th international conference on industrial and information systems (ICIIS)*, pages 1–5. IEEE, 2012.

[32] Yifan Cai and Simon X Yang. A survey on multi-robot systems. In *World Automation Congress 2012*, pages 1–6. IEEE, 2012.

[33] Ali Dorri, Salil S Kanhere, and Raja Jurdak. Multi-agent systems: A survey. *Ieee Access*, 6:28573–28593, 2018.

[34] Alessandro Farinelli, Luca Iocchi, and Daniele Nardi. Multirobot systems: a classification focused on coordination. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 34(5):2015–2028, 2004.

[35] Lauro Snidaro, J Garcia-Herrera, James Llinas, and Erik Blasch. Context-enhanced information fusion. *Boosting Real-World Performance with Domain Knowledge*, 2016.

[36] Cesar Cadena, Luca Carlone, Henry Carrillo, Yasir Latif, Davide Scaramuzza, José Neira, Ian Reid, and John J Leonard. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Transactions on robotics*, 32(6):1309–1332, 2016.

[37] F Kamil, S Tang, W Khaksar, N Zulkifli, and S Ahmad. A review on motion planning and obstacle avoidance approaches in dynamic environments. *Advances in Robotics & Automation*, 4(2):134–142, 2015.

[38] Emmanouil Tsardoulias and Pericles Mitkas. Robotic frameworks, architectures and middleware comparison. *arXiv preprint arXiv:1711.06842*, 2017.

[39] Claudine Badue, Rânik Guidolini, Raphael Vivacqua Carneiro, Pedro Azevedo, Vinicius B Cardoso, Avelino Forechi, Luan Jesus, Rodrigo Berriel, Thiago M Paixao, Filipe Mutz, et al. Self-driving cars: A survey. *Expert Systems with Applications*, 165:113816, 2021.

[40] Zhi Yan, Nicolas Jouandeau, and Arab Ali Cherif. A survey and analysis of multi-robot coordination. *International Journal of Advanced Robotic Systems*, 10(12):399, 2013.

[41] Janardan Kumar Verma and Virender Ranga. Multi-robot coordination analysis, taxonomy, challenges and future scope. *Journal of intelligent & robotic systems*, 102:1–36, 2021.

[42] Imad Jawhar, Nader Mohamed, Jie Wu, and Jameela Al-Jaroodi. Networking of multi-robot systems: architectures and requirements. *Journal of Sensor and Actuator Networks*, 7(4):52, 2018.

[43] Rajesh Doriya, Siddharth Mishra, and Swati Gupta. A brief survey and analysis of multi-robot communication and coordination. In *International conference on computing, communication & automation*, pages 1014–1021. IEEE, 2015.

[44] Stergios I Roumeliotis and George A Bekey. Distributed multirobot localization. *IEEE transactions on robotics and automation*, 18(5):781–795, 2002.

[45] Vladimir Zadorozhny and Michael Lewis. Information fusion based on collective intelligence for multi-robot search and rescue missions. In *2013 IEEE 14th International Conference on Mobile Data Management*, volume 1, pages 275–278. IEEE, 2013.

[46] Cyril Robin and Simon Lacroix. Multi-robot target detection and tracking: taxonomy and survey. *Autonomous Robots*, 40:729–760, 2016.

[47] Yugang Liu and Goldie Nejat. Robotic urban search and rescue: A survey from the control perspective. *Journal of Intelligent & Robotic Systems*, 72:147–165, 2013.

[48] Siobhán Grayson. Search & rescue using multi-robot systems. *School of Computer Science and Informatics, University College Dublin*, pages 1–14, 2014.

[49] Wanqing Zhao, Qinggang Meng, and Paul WH Chung. A heuristic distributed task allocation method for multivehicle multitask problems and its application to search and rescue scenario. *IEEE transactions on cybernetics*, 46(4):902–915, 2015.

[50] Tamer Abukhalil, Madhav Patil, Sarosh Patel, and Tarek Sobh. Coordinating a heterogeneous robot swarm using robot utility-based task assignment (ruta). In *2016 IEEE 14th International Workshop on Advanced Motion Control (AMC)*, pages 57–62. IEEE, 2016.

[51] David Portugal and Rui Rocha. A survey on multi-robot patrolling algorithms. In *Technological Innovation for Sustainability: Second IFIP WG 5.5/SOCOLNET Doctoral Conference on Computing, Electrical and Industrial Systems, DoCEIS 2011, Costa de Caparica, Portugal, February 21-23, 2011. Proceedings 2*, pages 139–146. Springer, 2011.

[52] Noa Agmon, Chien-Liang Fok, Yehuda Emaliah, Peter Stone, Christine Julien, and Sriram Vishwanath. On coordination in practical multi-robot patrol. In *2012 IEEE International Conference on Robotics and Automation*, pages 650–656. IEEE, 2012.

[53] David Portugal and Rui P Rocha. Distributed multi-robot patrol: A scalable and fault-tolerant framework. *Robotics and Autonomous Systems*, 61(12):1572–1587, 2013.

[54] Charles Pippin, Henrik Christensen, and Lora Weiss. Performance based task assignment in multi-robot patrolling. In *Proceedings of the 28th annual ACM symposium on applied computing*, pages 70–76, 2013.

[55] David Portugal and Rui P Rocha. Cooperative multi-robot patrol with bayesian learning. *Autonomous Robots*, 40(5):929–953, 2016.

[56] Alessandro Renzaglia, Lefteris Doitsidis, Agostino Martinelli, and Elias B Kosmatopoulos. Adaptive-based, scalable design for autonomous multi-robot surveillance. In *49th IEEE Conference on Decision and Control (CDC)*, pages 4618–4624. IEEE, 2010.

[57] Alejandro Pustowka and Eduardo F Caicedo. Market-based task allocation in a multi-robot surveillance system. In *2012 Brazilian Robotics Symposium and Latin American Robotics Symposium*, pages 185–189. IEEE, 2012.

[58] M Padmanabhan and GR Suresh. Coalition formation and task allocation of multiple autonomous robots. In *2015 3rd International Conference on Signal Processing, Communication and Networking (ICSCN)*, pages 1–5. IEEE, 2015.

[59] Dong-Hyun Lee, Ji-Hyeong Han, and Jong-Hwan Kim. A preference-based task allocation framework for multi-robot coordination. In *2011 IEEE International Conference on Robotics and Biomimetics*, pages 2925–2930. IEEE, 2011.

[60] Paula García, Pilar Caamaño, Richard J Duro, and Francisco Bellas. Scalable task assignment for heterogeneous multi-robot teams. *International journal of advanced robotic systems*, 10(2):105, 2013.

[61] Seohyun Jeon, Minsu Jang, Daeha Lee, Young-Jo Cho, and Jaeyeon Lee. Multiple robots task allocation for cleaning a large public space. In *2015 SAI Intelligent Systems Conference (IntelliSys)*, pages 315–319. IEEE, 2015.

[62] Daniel Claes, Frans Oliehoek, Hendrik Baier, Karl Tuyls, et al. Decentralised online planning for multi-robot warehouse commissioning. In *AAMAS'17: PROCEEDINGS OF THE 16TH INTERNATIONAL CONFERENCE ON AUTONOMOUS AGENTS AND MULTIAGENT SYSTEMS*, pages 492–500, 2017.

[63] Chayan Sarkar, Himadri Sekhar Paul, and Arindam Pal. A scalable multi-robot task allocation algorithm. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5022–5027. IEEE, 2018.

[64] Lars Johannsmeier and Sami Haddadin. A hierarchical human-robot interaction-planning framework for task allocation in collaborative industrial assembly processes. *IEEE Robotics and Automation Letters*, 2(1):41–48, 2016.

[65] Edoardo Lamon, Alessandro De Franco, Luka Peternel, and Arash Ajoudani. A capability-aware role allocation approach to industrial assembly tasks. *IEEE Robotics and Automation Letters*, 4(4):3378–3385, 2019.

[66] KS Senthilkumar and Kamal Kant Bharadwaj. Multi-robot exploration and terrain coverage in an unknown environment. *Robotics and Autonomous Systems*, 60(1):123–132, 2012.

[67] Robert Reid, Andrew Cann, Calum Meiklejohn, Liam Poli, Adrian Boeing, and Thomas Braunl. Cooperative multi-robot navigation, exploration, mapping and object detection with ros. In *2013 IEEE Intelligent Vehicles Symposium (IV)*, pages 1083–1088. IEEE, 2013.

[68] Xuefeng Dai, Laihao Jiang, and Yan Zhao. Cooperative exploration based on supervisory control of multi-robot systems. *Applied Intelligence*, 45:18–29, 2016.

[69] Pablo Iñigo-Blasco, Fernando Diaz-del Rio, Ma Carmen Romero-Ternero, Daniel Cagigas-Muñiz, and Saturnino Vicente-Diaz. Robotics software frameworks for

multi-agent robotic systems development. *Robotics and Autonomous Systems*, 60(6):803–821, 2012.

[70] Bo Chen, Harry H Cheng, and Joe Palen. Mobile-c: a mobile agent platform for mobile c/c++ agents. *Software: Practice and Experience*, 36(15):1711–1733, 2006.

[71] Bo Chen. *Runtime support for code mobility in distributed systems*. University of California, Davis, 2005.

[72] Wojciech Turek. Extensible multi-robot system. In *Computational Science–ICCS 2008: 8th International Conference, Kraków, Poland, June 23-25, 2008, Proceedings, Part III 8*, pages 574–583. Springer, 2008.

[73] Zhenglu Wang, Huaglory Tianfield, and Ping Jiang. A framework for coordination in multi-robot systems. In *IEEE International Conference on Industrial Informatics, 2003. INDIN 2003. Proceedings.*, pages 483–489. IEEE, 2003.

[74] Joao Frazao and Pedro Lima. Agent-based software architecture for multi-robot teams. *IFAC Proceedings Volumes*, 37(8):956–961, 2004.

[75] Chia-How Lin, Kai-Tai Song, and Gary T Anderson. Agent-based robot control design for multi-robot cooperation. In *2005 IEEE International Conference on Systems, Man and Cybernetics*, volume 1, pages 542–547. IEEE, 2005.

[76] Sanem Sariel-Talay, Tucker R Balch, and Nadia Erdogan. A generic framework for distributed multirobot cooperation. *Journal of Intelligent & Robotic Systems*, 63:323–358, 2011.

[77] Stephen Sodokan Nestinger. *A reconfigurable cooperative control system for rapid deployment of multi-robot systems*. University of California, Davis, 2009.

[78] Pedro M Shiroma and Mario FM Campos. Comutar: A framework for multi-robot coordination and task allocation. In *2009 IEEE/RSJ international conference on intelligent robots and systems*, pages 4817–4824. IEEE, 2009.

[79] David Vallejo, Paolo Remagnino, Dorothy N Monekosso, Luis Jiménez, and Carlos González. A multi-agent architecture for multi-robot surveillance. In *Computational Collective Intelligence. Semantic Web, Social Networks and Multiagent Systems: First International Conference, ICCCI 2009, Wrocław, Poland, October 5-7, 2009. Proceedings 1*, pages 266–278. Springer, 2009.

[80] Matthias Rambow, Florian Rohrmüller, Omiros Kourakos, Dirk WOLLHERR, Sandra HIRCHE, Martin BUSS, et al. A framework for information distribution, task execution and decision making in multi-robot systems. *IEICE TRANSACTIONS on Information and Systems*, 93(6):1352–1360, 2010.

[81] Joaquin López, Diego Pérez, and Eduardo Zalama. A framework for building mobile single and multi-robot applications. *Robotics and Autonomous Systems*, 59(3-4):151–162, 2011.

[82] Binsen Qian. *A Mobile Agent-based Framework for Automatic Coordination of Autonomous Multi-robot Systems*. University of California, Davis, 2018.

[83] M Oprea. Agent-based modelling of multi-robot systems. In *IOP Conference Series: Materials Science and Engineering*, volume 444, page 052026. IOP Publishing, 2018.

[84] Ahmed R Sadik, Christian Goerick, and Manuel Muehlig. Modeling and simulation of a multi-robot system architecture. In *2019 International Conference on Mechatronics, Robotics and Systems Engineering (MoRSE)*, pages 8–14. IEEE, 2019.

[85] Yara Rizk, Mariette Awad, and Edward W Tunstel. Cooperative heterogeneous multi-robot systems: A survey. *ACM Computing Surveys (CSUR)*, 52(2):1–31, 2019.

[86] Luis C Cobo, Charles L Isbell Jr, and Andrea L Thomaz. Automatic task decomposition and state abstraction from demonstration. Georgia Institute of Technology, 2012.

[87] Barbara Arbanas, Antun Ivanovic, Marko Car, Tomislav Haus, Matko Orsag, Tamara Petrovic, and Stjepan Bogdan. Aerial-ground robotic system for autonomous delivery tasks. In *2016 IEEE international conference on robotics and automation (ICRA)*, pages 5463–5468. IEEE, 2016.

[88] K Vinh, Solomon Gebreyohannes, and Ali Karimoddini. An area-decomposition based approach for cooperative tasking and coordination of uavs in a search and coverage mission. In *2019 IEEE Aerospace Conference*, pages 1–8. IEEE, 2019.

[89] G Ayorkor Korsah, Anthony Stentz, and M Bernardine Dias. A comprehensive taxonomy for multi-robot task allocation. *The International Journal of Robotics Research*, 32(12):1495–1512, 2013.

[90] Brian P Gerkey and Maja J Matarić. A formal analysis and taxonomy of task allocation in multi-robot systems. *The International journal of robotics research*, 23(9):939–954, 2004.

[91] BB Choudhury and BB Biswal. Task allocation methodologies for multi-robot systems. 2008.

[92] Alejandro R Mosteo and Luis Montano. A survey of multi-robot task allocation. *Instituto de Investigacin en Ingenierła de Aragn (I3A), Tech. Rep*, 2010.

[93] Seenu N, Kuppan Chetty RM, Ramya MM, and Mukund Nilakantan Janardhanan. Review on state-of-the-art dynamic task allocation strategies for multiple-robot systems. *Industrial Robot: the international journal of robotics research and application*, 47(6):929–942, 2020.

[94] Lynne E Parker. Alliance: An architecture for fault tolerant multirobot cooperation. *IEEE transactions on robotics and automation*, 14(2):220–240, 1998.

[95] Lynne E Parker. L-alliance: A mechanism for adaptive action selection in heterogeneous multi-robot teams. Technical report, Oak Ridge National Lab.(ORNL), Oak Ridge, TN (United States), 1995.

[96] Jianping Wang, Yuesheng Gu, and Xiaomin Li. Multi-robot task allocation based on ant colony algorithm. *J. Comput.*, 7(9):2160–2167, 2012.

[97] Zhi Li, Ali Vatankhah Barenji, Jiazhi Jiang, Ray Y Zhong, and Gangyan Xu. A mechanism for scheduling multi robot intelligent warehouse system face with dynamic demand. *Journal of Intelligent Manufacturing*, 31:469–480, 2020.

[98] E Gil Jones, M Bernardine Dias, and Anthony Stentz. Time-extended multi-robot coordination for domains with intra-path constraints. *Autonomous robots*, 30:41–56, 2011.

[99] Alejandro R Mosteo and Luis Montano. Simulated annealing for multi-robot hierarchical task allocation with flexible constraints and objective functions. In *Workshop on network robot systems: toward intelligent robotic systems integrated with environments. int. conf. on intelligent robots and systems*. Citeseer, 2006.

[100] Alaa Khamis, Ahmed Hussein, and Ahmed Elmogy. Multi-robot task allocation: A review of the state-of-the-art. *Cooperative robots and sensor networks 2015*, pages 31–51, 2015.

[101] M Bernardine Dias, Robert Zlot, Nidhi Kalra, and Anthony Stentz. Market-based multirobot coordination: A survey and analysis. *Proceedings of the IEEE*, 94(7):1257–1270, 2006.

[102] Ahmed Hussein, Mohamed Adel, Mohamed Bakr, Omar M Shehata, and Alaa Khamis. Multi-robot task allocation for search and rescue missions. In *Journal of Physics: Conference Series*, volume 570, page 052006. IOP Publishing, 2014.

[103] Jieke Shi, Zhou Yang, and Junwu Zhu. An auction-based rescue task allocation approach for heterogeneous multi-robot system. *Multimedia Tools and Applications*, 79:14529–14538, 2020.

[104] Mohamed Badreldin, Ahmed Hussein, and Alaa Khamis. A comparative study between optimization and market-based approaches to multi-robot task allocation. *Advances in Artificial Intelligence*, 2013:12–12, 2013.

[105] Lingzhi Luo, Nilanjan Chakraborty, and Katia Sycara. Competitive analysis of repeated greedy auction algorithm for online multi-robot task assignment. In *2012 IEEE International Conference on Robotics and Automation*, pages 4792–4799. IEEE, 2012.

[106] M Bernardine Dias. *Traderbots: A new paradigm for robust and efficient multirobot coordination in dynamic environments*. Carnegie Mellon University, 2004.

[107] M Bernardine Dias and Anthony Stentz. Opportunistic optimization for market-based multirobot control. In *IEEE/RSJ international conference on intelligent robots and systems*, volume 3, pages 2714–2720. IEEE, 2002.

[108] Michael Otte, Michael J Kuhlman, and Donald Sofge. Auctions for multi-robot task allocation in communication limited environments. *Autonomous Robots*, 44:547–584, 2020.

[109] Han-Lim Choi, Luc Brunet, and Jonathan P How. Consensus-based decentralized auctions for robust task allocation. *IEEE transactions on robotics*, 25(4):912–926, 2009.

[110] Donato Di Paola, David Naso, and Biagio Turchiano. Consensus-based robust decentralized task assignment for heterogeneous robot networks. In *Proceedings of the 2011 American Control Conference*, pages 4711–4716. IEEE, 2011.

[111] Kai Zhang, Emmanuel G Collins Jr, and Dongqing Shi. Centralized and distributed task allocation in multi-robot teams via a stochastic clustering auction. *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, 7(2):1–22, 2012.

[112] Kristina Lerman, Chris Jones, Aram Galstyan, and Maja J Matarić. Analysis of dynamic task allocation in multi-robot systems. *The International Journal of Robotics Research*, 25(3):225–241, 2006.

[113] Sameera Ponda, Josh Redding, Han-Lim Choi, Jonathan P How, Matt Vavrina, and John Vian. Decentralized planning for complex missions with dynamic communication constraints. In *Proceedings of the 2010 American Control Conference*, pages 3998–4003. IEEE, 2010.

[114] Athanasios Tsalatsanis, Ali Yalcin, and Kimon P Valavanis. Dynamic task allocation in cooperative robot teams. *Robotica*, 30(5):721–730, 2012.

[115] Charles E Pippin and Henrik Christensen. A bayesian formulation for auction-based task allocation in heterogeneous multi-agent teams. In *Ground/Air Multisensor Interoperability, Integration, and Networking for Persistent ISR II*, volume 8047, pages 242–252. SPIE, 2011.

[116] Weiyu Wang and Keng Siau. Artificial intelligence, machine learning, automation, robotics, future of work and future of humanity: A review and research agenda. *Journal of Database Management (JDM)*, 30(1):61–79, 2019.

[117] Batta Mahesh. Machine learning algorithms-a review. *International Journal of Science and Research (IJSR).[Internet]*, 9:381–386, 2020.

[118] Stuart J Russell. *Artificial intelligence a modern approach*. Pearson Education, Inc., 2010.

[119] Jafar Alzubi, Anand Nayyar, and Akshi Kumar. Machine learning from theory to algorithms: an overview. In *Journal of physics: conference series*, volume 1142, page 012012. IOP Publishing, 2018.

[120] K Sindhu Meena and S Suriya. A survey on supervised and unsupervised learning techniques. In *Proceedings of International Conference on Artificial Intelligence, Smart Grid and Smart City Applications: AISGSC 2019*, pages 627–644. Springer, 2020.

[121] Ajay Shrestha and Ausif Mahmood. Review of deep learning algorithms and architectures. *IEEE access*, 7:53040–53065, 2019.

[122] NV Balaji and M Punithavalli. Machine learning approach for object detection-a survey approach. *International Journal of Computer Science and Information Security*, 8(7):67–71, 2010.

[123] Zhong-Qiu Zhao, Peng Zheng, Shou-tao Xu, and Xindong Wu. Object detection with deep learning: A review. *IEEE transactions on neural networks and learning systems*, 30(11):3212–3232, 2019.

[124] Anamika Dhillon and Gyanendra K Verma. Convolutional neural network: a review of models, methodologies and applications to object detection. *Progress in Artificial Intelligence*, 9(2):85–112, 2020.

[125] Balasubramaniyan Chandrasekaran, Shruti Gangadhar, and James M Conrad. A survey of multisensor fusion techniques, architectures and methodologies. In *SoutheastCon 2017*, pages 1–8. IEEE, 2017.

[126] Maria Alessandra Montironi. *An Attention-based Methodology for Context Identification and Exploitation in Autonomous Robots*. PhD thesis, University of California, Davis, 2018.

[127] Stamatios Samaras, Eleni Diamantidou, Dimitrios Ataloglou, Nikos Sakellariou, Anastasios Vafeiadis, Vasilis Magoulianitis, Antonios Lalas, Anastasios Dimou, Dimitrios Zarpalas, Konstantinos Votis, et al. Deep learning on multi sensor data for counter uav applications—a systematic review. *Sensors*, 19(22):4837, 2019.

[128] Tong Meng, Xuyang Jing, Zheng Yan, and Witold Pedrycz. A survey on machine learning for data fusion. *Information Fusion*, 57:115–129, 2020.

[129] Hongjun Zhou and Shigeyuki Sakane. Sensor planning for mobile robot localization—a hierarchical approach using a bayesian network and a particle filter. *IEEE Transactions on Robotics*, 24(2):481–487, 2008.

[130] Lingwen Zhang, Yishun Li, Yajun Gu, and Wenkao Yang. An efficient machine learning approach for indoor localization. *China Communications*, 14(11):141–150, 2017.

[131] Daoud Burghal, Ashwin T Ravi, Varun Rao, Abdullah A Alghafis, and Andreas F Molisch. A comprehensive survey of machine learning based localization with wireless signals. *arXiv preprint arXiv:2012.11171*, 2020.

[132] KM Akhil and Somnath Sinha. Self-localization in large scale wireless sensor network using machine learning. In *2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE)*, pages 1–5. IEEE, 2020.

[133] Ying Wang and Clarence W de Silva. A machine-learning approach to multi-robot coordination. *Engineering Applications of Artificial Intelligence*, 21(3):470–484, 2008.

[134] Hongliang Guo and Yan Meng. Distributed reinforcement learning for coordinate multi-robot foraging. *Journal of intelligent & robotic systems*, 60:531–551, 2010.

[135] Xinyi Zhao, Qun Zong, Bailing Tian, Boyuan Zhang, and Ming You. Fast task allocation for heterogeneous unmanned aerial vehicles through reinforcement learning. *Aerospace Science and Technology*, 92:588–594, 2019.

[136] Wei Dai, Huimin Lu, Junhao Xiao, Zhiwen Zeng, and Zhiqiang Zheng. Multi-robot dynamic task allocation for exploration and destruction. *Journal of Intelligent & Robotic Systems*, 98:455–479, 2020.

[137] George Marios Skaltsis, Hyo-Sang Shin, and Antonios Tsourdos. A survey of task allocation techniques in mas. In *2021 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 488–497. IEEE, 2021.

[138] Yong-yan Hou and Wen-Qiang Guo. Optimal coordination of multi-task allocation and path planning for uavs using dynamic bayesian network. In *2009 Chinese Control and Decision Conference*, pages 3590–3594. IEEE, 2009.

[139] Gregory Kahn, Adam Villaflor, Vitchyr Pong, Pieter Abbeel, and Sergey Levine. Uncertainty-aware reinforcement learning for collision avoidance. *arXiv preprint arXiv:1702.01182*, 2017.

[140] Xiaoyun Lei, Zhian Zhang, and Peifang Dong. Dynamic path planning of unknown environment based on deep reinforcement learning. *Journal of Robotics*, 2018, 2018.

[141] Bradley Woosley and Prithviraj Dasgupta. Integrated real-time task and motion planning for multiple robots under path and communication uncertainties. *Robotica*, 36(3):353–373, 2018.

[142] Dan Song, Carl Henrik Ek, Kai Huebner, and Danica Kragic. Multivariate discretization for bayesian network structure learning in robot grasping. In *2011 IEEE International Conference on Robotics and Automation*, pages 1944–1950. IEEE, 2011.

[143] Dan Song, Carl Henrik Ek, Kai Huebner, and Danica Kragic. Task-based robot grasp planning using probabilistic inference. *IEEE transactions on robotics*, 31(3):546–561, 2015.

[144] Felix Spenrath and Andreas Pott. Using neural networks for heuristic grasp planning in random bin picking. In *2018 IEEE 14th International Conference on Automation Science and Engineering (CASE)*, pages 258–263. IEEE, 2018.

[145] Andy Zeng, Shuran Song, Stefan Welker, Johnny Lee, Alberto Rodriguez, and Thomas Funkhouser. Learning synergies between pushing and grasping with self-supervised deep reinforcement learning. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4238–4245. IEEE, 2018.

[146] Kilian Kleeberger, Richard Bormann, Werner Kraus, and Marco F Huber. A survey on learning-based robotic grasping. *Current Robotics Reports*, 1:239–249, 2020.

[147] Mennatallah Amer, Markus Goldstein, and Slim Abdennadher. Enhancing one-class support vector machines for unsupervised anomaly detection. In *Proceedings of the ACM SIGKDD workshop on outlier detection and description*, pages 8–15, 2013.

[148] Marcelo Azevedo Costa, Bernhard Wullt, Mikael Norrlöf, and Svante Gunnarsson. Failure detection in robotic arms using statistical modeling, machine learning and hybrid gradient boosting. *Measurement*, 146:425–436, 2019.

[149] Riccardo Pinto and Tania Cerquitelli. Robot fault detection and remaining life estimation for predictive maintenance. *Procedia Computer Science*, 151:709–716, 2019.

[150] Zoubin Ghahramani. Probabilistic machine learning and artificial intelligence. *Nature*, 521(7553):452–459, 2015.

[151] Susmita Ray. A quick review of machine learning algorithms. In *2019 International conference on machine learning, big data, cloud and parallel computing (COMITCon)*, pages 35–39. IEEE, 2019.

[152] Sotiris B Kotsiantis, Ioannis Zaharakis, P Pintelas, et al. Supervised machine learning: A review of classification techniques. *Emerging artificial intelligence applications in computer engineering*, 160(1):3–24, 2007.

[153] Jens Kober, J Andrew Bagnell, and Jan Peters. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 32(11):1238–1274, 2013.

[154] Niko Sünderhauf, Oliver Brock, Walter Scheirer, Raia Hadsell, Dieter Fox, Jürgen Leitner, Ben Upcroft, Pieter Abbeel, Wolfram Burgard, Michael Milford, et al. The limits and potentials of deep learning for robotics. *The International journal of robotics research*, 37(4-5):405–420, 2018.

[155] Christopher M Bishop. Model-based machine learning. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 371(1984):20120222, 2013.

[156] Adnan Darwiche. *Modeling and reasoning with Bayesian networks*. Cambridge university press, 2009.

[157] Han-Saem Park and Sung-Bae Cho. A modular design of bayesian networks using expert knowledge: Context-aware home service robot. *Expert Systems with Applications*, 39(3):2629–2642, 2012.

[158] Xiao Jia and Max Q-H Meng. A survey and analysis of task allocation algorithms in multi-robot systems. In *2013 IEEE International Conference on Robotics and biomimetics (ROBIO)*, pages 2280–2285. IEEE, 2013.

[159] Laura Von Rueden, Sebastian Mayer, Katharina Beckh, Bogdan Georgiev, Sven Giesselbach, Raoul Heese, Birgit Kirsch, Julius Pfrommer, Annika Pick, Rajkumar Ramamurthy, et al. Informed machine learning–a taxonomy and survey of integrating prior knowledge into learning systems. *IEEE Transactions on Knowledge and Data Engineering*, 35(1):614–633, 2021.

[160] Liang Yang, Juntong Qi, Dalei Song, Jizhong Xiao, Jianda Han, Yong Xia, et al. Survey of robot 3d path planning algorithms. *Journal of Control Science and Engineering*, 2016, 2016.

[161] Mahdi Fakoor, Amirreza Kosari, and Mohsen Jafarzadeh. Revision on fuzzy artificial potential field for humanoid robot path planning in unknown environment. *International Journal of Advanced Mechatronic Systems*, 6(4):174–183, 2015.

[162] Daniel Kersten, Pascal Mamassian, and Alan Yuille. Object perception as bayesian inference. *Annu. Rev. Psychol.*, 55:271–304, 2004.

[163] Stergios I Roumeliotis and George A Bekey. Extended kalman filter for frequent local and infrequent global sensor data fusion. In *Sensor Fusion and Decentralized Control in Autonomous Robotic Systems*, volume 3209, pages 11–22. SPIE, 1997.

[164] Hao Li and Fawzi Nashashibi. Cooperative multi-vehicle localization using split covariance intersection filter. *IEEE Intelligent transportation systems magazine*, 5(2):33–44, 2013.

[165] Thumeera R Wanasinghe, George KI Mann, and Raymond G Gosine. Decentralized cooperative localization for heterogeneous multi-robot system using split covariance intersection filter. In *2014 Canadian Conference on Computer and Robot Vision*, pages 167–174. IEEE, 2014.

[166] Joelle Al Hage, Maan E El Najjar, and Denis Pomorski. Multi-sensor fusion approach with fault detection and exclusion based on the kullback–leibler divergence:

Application on collaborative multi-robot system. *Information Fusion*, 37:61–76, 2017.

[167] Jesus Capitan, Matthijs TJ Spaan, Luis Merino, and Anibal Ollero. Decentralized multi-robot cooperation with auctioned pomdps. *The International Journal of Robotics Research*, 32(6):650–671, 2013.

[168] Ching-Wei Chuang and Harry H Cheng. A novel approach with bayesian networks to multi-robot task allocation in dynamic environments. In *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, volume 85444, page V08AT08A047. American Society of Mechanical Engineers, 2021.

[169] Ching-Wei Chuang. The videos of the hardware experiments in the dissertation, 2023. `https://ucdavis.box.com/s/rdc6a0mteq431gja1oide7s0xz6okzdn`.