Human-Machine Collaborative Optimization via Apprenticeship Scheduling

by

Matthew C. Gombolay

Submitted to the Department of Aeronautics and Astronautics
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy in Autonomous Systems
at the
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
February 2017

Author . . . .
Department of Aeronautics and Astronautics
January 11, 2017

Certified by . . . .
Prof. Julie Shah
Thesis Supervisor

Certified by . . . .
Prof. Andrea Thomaz
Thesis Committee Member

Certified by . . . .
Prof. Bilge Mutlu
Thesis Committee Member

Certified by . . . .
Prof. Hamsa Balakrishnan
Thesis Committee Member

Certified by . . . .
Prof. Peter Szolovits
Thesis Committee Member

Accepted by . . . .
Youssef M. Marzouk
Associate Professor of Aeronautics and Astronautics
Chair, Graduate Program Committee

Human-Machine Collaborative Optimization via Apprenticeship Scheduling

by

Matthew C. Gombolay

Submitted to the Department of Aeronautics and Astronautics
on January 11, 2017, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Autonomous Systems

Abstract

I envision a future where intelligent service robots become integral members of human-robot teams in the workplace. Today, service robots are being deployed across a wide range of settings; however, while these robots exhibit basic navigational abilities, they lack the ability to anticipate and adapt to the needs of their human teammates. I believe robots must be capable of autonomously learning from humans how to integrate into a team à la a human apprentice.

Human domain experts and professionals become experts over years of apprenticeship, and this knowledge is not easily codified in the form of a policy. In my thesis, I develop a novel computational technique, Collaborative Optimization Via Apprenticeship Scheduling (COVAS), that enables robots to learn a policy to capture an expert's knowledge by observing the expert solve scheduling problems. COVAS can then leverage the policy to guide branch-and-bound search to provide globally optimal solutions faster than state-of-the-art optimization techniques.

Developing an apprenticeship learning technique for scheduling is challenging because of the complexities of modeling and solving scheduling problems. Previously, researchers have sought to develop techniques to learn from human demonstration; however, these approaches have rarely been applied to scheduling because of the large number of states required to encode the possible permutations of the problem and relevant problem features (e.g., a job's deadlines, required resources, etc.).

My thesis gives robots a novel ability to serve as teammates that can learn from and contribute to coordinating a human-robot team. The key to COVAS' ability to efficiently and optimally solve scheduling problems is the use of a novel policy-learning approach – apprenticeship scheduling – suited for imitating the method an expert uses to generate the schedule. This policy learning technique uses pairwise comparisons between the action taken by a human expert (e.g., schedule agent a to complete task τ_i at time t) and each action not taken (e.g., unscheduled tasks at time t), at each moment in time, to learn the relevant model parameters and scheduling policies demonstrated in training examples provided by the human experts.

I evaluate my technique in two real-world domains. First, I apply apprenticeship scheduling to the problem of anti-ship missile defense: protecting a naval vessel from an enemy attack by deploying decoys and countermeasures at the right place and time. I show that apprenticeship scheduling can learn to defend the ship, outperforming human experts on the majority of naval engagements ($p < 0.011$). Further, COVAS is able to produce globally optimal solutions an order of magnitude faster than traditional, state-of-the-art optimization techniques. Second, I apply apprenticeship scheduling to learn how to function as a resource nurse: the nurse in charge of ensuring the right patient is in the right type of room at the right time and that the right types of nurses are there to care for the patient. After training an apprentice scheduler on demonstrations given by resource nurses, I found that nurses and physicians agreed with the algorithm's advice 90% of the time.

Next, I conducted a series of human-subject experiments to understand the human factors consequences of embedding scheduling algorithms in robotic platforms. Through these experiments, I found that an embodied platform (i.e., a physical robot) engenders more appropriate trust and reliance in the system than an un-embodied one (i.e., computer-based system) when the scheduling algorithm works with human domain experts. However, I also found that increasing robot autonomy degrades human situational awareness. Further, there is a complex interplay between workload and workflow preferences that must be balanced to maximize team fluency. Based on these findings, I develop design guidelines for integrating service robots with autonomous decision-making capabilities into the human workplace.

Prof. Julie Shah
Thesis Supervisor

Prof. Andrea Thomaz
Thesis Committee Member

Prof. Bilge Mutlu
Thesis Committee Member

Prof. Hamsa Balakrishnan
Thesis Committee Member

Prof. Peter Szolovits
Thesis Committee Member

Dr. Laura Hiatt
Thesis Reader

Dr. Sung-Hyun Son
Thesis Reader

Acknowledgements

I would like to take the opportunity to acknowledge the contributions of those who have supported me in my quest to defending a doctorate of philosophy.

In early 2011, the graduate program administrator in the Department of Aeronautics and Astronautics at the Massachusetts Institute of Technology contacted me to inform me that I had been admitted to the program and that Professor Julie Shah would like to bring me into her research lab. I remember the first phone call I ever had with Julie. Of all the professors I spoke to about the awesome work I could do with them, she was the one that inspired me the most. After visiting MIT during the AeroAstro visit day, meeting Julie, and getting a tour of the lab,[1] I knew I had found my home. I knew Julie was the mentor I wanted to train under for my PhD. Approaching the completion of my dissertation, I can now say that chatting with Julie brings me even more inspiration than it did five and a half years ago. She is incredibly intelligent, driven, and one of the most focused listeners I have ever known. Julie has molded, shaped, and formed me into the researcher that I am today. She pushed me when I needed to be pushed, and she slowed me when I needed to be slowed. She taught me how to do rigorous research, good science, and disseminate my work to a broad community. I would not be the academic I am today without her time, effort, and care. Julie, thank you – I am eternally grateful.

I would like to thank all of my thesis committee members, Andrea Thomaz, Bilge Mutlu, Hamsa Balakrishnan, and Peter Szolovits. I admire each and every one of them. Andrea and Bilge were two of the very first professors I met in the field of

[1] At that time, the lab was a room full of ancient turbomachinery, but I saw Julie's vision for how that space could be transformed into an awesome robotics lab.

human-robot interaction. When I entered the turbulent waters of academia, Andrea and Bilge welcomed me and adopted me as their own. I still remember pitching my thesis ideas to Andrea, and she said that she totally got it. In her wisdom, Andrea guided me toward making the right assumptions in my work that helped me head down a fruitful research path. She has been an incredible mentor, helping me in my professional development and grow as a member of the academic community. I also remember meeting Bilge at RSS'13. He encouraged and challenged me to look beyond my work and carefully consider how to design the interaction of a human-robot team. He pushed me to improve my experiment design, isolate the key and confounding variables, and think bigger. Much of the inspiration for my thesis comes from Hamsa Balakrishnan. Hamsa's work on modeling air traffic controllers emboldened me to believe that scheduling models of domain experts can, in fact, be developed and used to improve decision making. Her perceptiveness and intuition has pushed me to understand the strengths and weakness of my work, as well as which challenge to tackle next. Hamsa has helped me to learn empathy, putting myself in the shoes of my audience to better understand how to conduct and communicate good research. Pete Szolovits has been my anchor to the world of medically-inspired artificial intelligence research. Pete has an incredible amount of experience and insight into which ideas have been tried, which have been left to try, and why. My chats with Pete are some of my favorite; I always leave feeling as though he has made me smarter. Pete has been kind, supportive, and tough on my research, and I am grateful to have gotten to know him.

I would also like to thank my thesis readers: Sung-Hyun Son and Laura Hiatt. One and a half years ago, Sung invited me to join him at MIT Lincoln Laboratory and see what apprenticeship scheduling could do for helping to train naval officers. He gave me a rare opportunity to work with real data of real human operators, and he gave me the freedom to brainstorm and explore my ideas. I admire his choice as a supervisor to implicitly trust me, and his confidence enabled me to succeed. I am grateful for his honest, candid, and supportive counsel. When I first met Laura, I knew that she was bright, supportive, and that I could learn a lot from her. I enjoy

brainstorming and debating ideas with her. I admire that her responses are insightful, challenging, and encouraging.

Thank you to Dr. Neel Shah, Dr. Toni Golen, and all of the nurses, physicians, medical students, and support staff in the OBGYN department at Beth Israel Deaconess Medical Center who adopted me as own of their one for the last few years. Neel mentored me as though I were a third year medical student. He showed me how to round on patients and write "SOAP" notes; he allowed me to shadow him while he treated patients and performed vaginal and cesarean deliveries; and he inspired me to passionately care for the health of women and their babies. Toni is a wonderful force for good in labor and delivery. If I were a woman giving birth, I would want her to be my second opinion. Toni helped to get me access to the people and information I needed to succeed. I admire her intellect and serious regard for her practice.

My time on labor and delivery would certainly have not been a success were it not for the amazing, brave, and dedicated nurses. They are a tight-knit family, dedicated to bringing new life into this world, and I feel privileged to have been trusted and accepted by them. I wish I could name them all here as a testament to their contribution to my thesis – they deserve so much credit. In particular, the charge nurses, who are responsible for running air traffic control operations for a labor and delivery floor, have a tremendously difficult, stressful job, with little reward. Yet, without them, the labor floor – any ward for that matter – would completely fall apart. My goal in applying apprenticeship scheduling to healthcare was to help improve the nurses' abilities to manage the logistics of labor and delivery and focus more on what they get up in the morning to do: heal. Almost every day I came on the floor, they asked me, "Matt, where's my robot?" I would typically laugh and say, "I'm working on it!" Nurses and physicians are some of the most over-worked, under-appreciated people I know. Anything we can do to help maximize the use of their time with better logistics would be a win-win for everyone. I hope that this thesis inspires other engineers to develop new technologies to support nurses and physicians.

I want to thank everyone who made the Interactive Robotics Group (IRG) great. IRG was my MIT home for five and a half years; I made many friends and more good

memories. Thanks first to Ron Wilcox. He showed me it was, in fact, possible to work together with someone on a team as an equal. Of all my time at MIT, learning it was possible was, perhaps, one of the most valuable lessons I received. I remember our week of late nights together getting a robot demonstration going, evenings shooting pool, and watching Power Rangers on Friday afternoons. Ron was also the one who developed an Excel GUI for the Tercio algorithm; I used this GUI hundreds of times to generate schedules for the human-robot teams I studied in my experiments. Thanks to Jessie Yang for helping me to craft an airtight case for the importance of the work in our RSS'16 paper. I also want to thank Brad Hayes for dropping everything for me at a moments notice for me to help get out an RSS paper in January 2016. Without him, the robot demonstration would not have happened in time, and I am grateful for his sacrifice to help me get it working in the eleventh hour.

Part of what makes IRG so special is the many, amazing undergraduates who join us to make robots do awesome things. I have greatly appreciated the opportunity to work with many of them during my time as a graduate student. Thank you to Michael Kelessoglou, Anna Bair, Tania Yu, Samir Wadwhania, Zixi "Zeo" Liu, Cindy Huang, Nicole, Seo, John Peurifoy, Eric Huppert, Giancarlo Sturla, Alex Gutierrez, Charlotte Kiang, Fei Yu, and Bryan Collazo. They have helped me to grow as a researcher, mentor, and leader. I am grateful for them entrusting their time to me and inspiring me to dream big. As it relates to this thesis, there are a few undergraduates I want to thank in particular. Thanks to Nicole for making transforming the ideas and blueprints I had for a labor floor simulation into reality. I used this simulation to collect data from resource nurses train and validate the utility of my machine learning algorithm, and I used it as the basis for a human-subject experiment in which nurses would receive advice on how to manage patients on the simulated labor floor. Nicole, you were a fantastic partner, and I know you are going to do amazing things. Thank you for all that you contributed to my endeavor. Thanks to Cindy Huang for helping me conduct human-subject experiments. Thanks Anna Bair for being incredibly responsible and teachable - you honored the trust I gave you by single-handedly executing an experimental protocol with real human subjects and a

PR2. Bravo. Thanks to Alex Gutierrez for coding up a control interface so that I could get the PR2 to pick up and drop off the Lego kits in my experiments. Thanks to Tania, Samir, and Zeo for working to build up the codebase for the NAO healthcare robot for the RSS'16 paper.

Next, I would like to acknowledge the contribution of my sponsors. This work was supported by the National Science Foundation Graduate Research Fellowship Program under grant number 2388357, the CRICO Harvard Risk Management Foundation, Aldebaran Robotics, and the US Navy (PMR-51).

Now I come to a more personal note. Thank you, Mom (Lauren) and Dad (Craig), for nurturing, raising, and setting for me a wonderful example to follow. Isaac Newton once said, "If I have seen further, it is by standing on the shoulders of giants." My parents provided me the tallest, most stable shoulders to stand on any child could ask for. Thank you for giving me so many wonderful opportunities to explore my passions and realize my dreams. My successes are your successes. I know you love me very much. Thank you, Alli, my dear, little sister, for loving and supporting me. You inspire me to be the best big brother I can be for you. Thank you all for all of your support as my editors-in-chief. Thank you for your prayers, love, and support.

Thank you to my beloved wife, Grace. You have lived my triumphs and my failures. You have known my joys and tragedies. I have no better friend on Earth (and one day Mars) than you. God has so deeply blessed me by giving me you, my best friend. In the most wonderfully paradoxical way, you make me more me. Every day I spend with you is a day I know that I am loved. Thank you for being my steadfast counsel, caregiver, and companion. Thank you for helping me prepare for my PhD qualifying exam. I vividly remember those hours going through slides with you. I was, perhaps, one of your most obstinate patients. Thank you for your patience, love, and friendship. You will always be my bride, and I will always say, "I do."

For my ultimate personal acknowledgment, I want to thank God. Nine years ago, I began a journey with Jesus Christ to discover the truth. It is written, "If you abide in my word, you are truly my disciples, and you will know the truth, and the truth

will set you free." I have always been drawn to the study of robotics and Artificial Intelligence. For me, developing computational methods has been an exploration of how I think and how my mind works – a mind that was given to me by him who "knitted me together in my mother's womb." As a research community, every challenge we face and discovery we make instills in me even more appreciation for the LORD's infinite knowledge, which created the heavens and earth. This thesis is his, and I rest in him.

Contents

1 Introduction **19**

1.1 Apprenticeship Scheduling: Learning to Schedule from Human Experts 26

1.2 Learning to Make Super-Human Scheduling Decisions 31

1.3 Robot Embodiment as a Scheduling Apprentice 34

1.4 Situational Awareness, Workload, and Workflow Preferences with Service Robots 38

2 Background **43**

2.1 Introduction 43

2.2 AI: Scheduling 46

2.2.1 MILP/CP Solution Techniques 47

2.2.2 Auction, Market-Based, and Heuristic Solution Techniques . . 50

2.2.3 Hybrid Solution Techniques 52

2.2.4 Conclusion 54

2.3 Learning from Human Demonstration 54

2.3.1 Recommender/Preference-Learning Systems 56

2.3.2 Challenges with Reward/Preference Learning 59

2.3.3 Policy/Imitation Learning 60

2.3.4 Conclusion 61

2.4 Blending Machine Learning and Optimization 62

2.4.1 Conclusion 64

2.5 Human Factors of Human-Robot Interaction 65

2.5.1 Human-In-The-Loop Decision Making 65

2.5.2 Aiding Humans via Autonomy 66
2.5.3 Situational Awareness 68
2.5.4 Mental and Physical Workload 70
2.5.5 Workflow/Scheduling Preferences 71
2.5.6 Embodiment 72
2.5.7 Conclusion 74

3 Apprenticeship Scheduling: Learning to Schedule from Human Experts 75

3.1 Introduction 75
3.2 Problem Domains 77
3.2.1 Synthetic Data Set 78
3.2.2 Anti-Ship Missile Defense 80
3.2.3 Labor and Delivery 86
3.3 Model for Apprenticeship Learning 90
3.4 Data Sets 96
3.4.1 Synthetic Data Set 96
3.4.2 Real-World Data Set: ASMD 99
3.4.3 Real World Dataset: Labor and Delivery 101
3.5 Empirical Evaluation 101
3.5.1 Synthetic Data Set 103
3.5.2 Real-World Data Set: ASMD 110
3.5.3 Real-World Data Set: Labor and Delivery 111
3.6 Anomalies and Future Work 111
3.7 Conclusions 113

4 Learning to Make Super-Human Scheduling Decisions 115

4.1 Introduction 115
4.2 Model for Collaborative Optimization via Apprenticeship Scheduling 116
4.2.1 COVAS Architecture 116
4.2.2 Apprenticeship Scheduling Subroutine 118

4.2.3 Optimization Subroutine 119
4.3 Training Dataset 120
4.4 Results and Discussion 121
4.4.1 Validation Against Expert Benchmark 121
4.4.2 Limitations and Future Work 122
4.5 Conclusions 123

5 Robot Embodiment as a Scheduling Apprentice 125
5.1 Introduction 125
5.2 Experimental Investigation 127
5.2.1 Hypotheses and Measures 128
5.2.2 Materials and Setup 129
5.2.3 Experimental Procedure 129
5.3 Implementation of Decision Support 130
5.4 Results 132
5.4.1 Analysis & Discussion of **H1** 132
5.4.2 Analysis & Discussion of **H2** 135
5.4.3 Discussion on Embodiment 137
5.5 Pilot Demonstration of a Robotic Assistant on the Labor and Delivery Floor 139
5.5.1 Robot System Architecture 140
5.6 Conclusion 143

6 Situational Awareness, Workload, and Workflow Preferences with Service Robots 145
6.1 Introduction 145
6.2 Aims of the Experiment 147
6.2.1 Experiment: Situational Awareness in Mixed-Initiative Human-Robot Teaming 148
6.2.2 Experiment: Workflow Preferences 153
6.2.3 Experiment: Workload vis á vis Workflow Preferences 155

6.3 Formal Problem Definition 158
6.4 Scheduling Mechanism 160
6.4.1 Algorithm Modifications for Mixed-Initiative Scheduling . . . 162
6.4.2 Algorithm Modifications for Scheduling with Preferences . . . 163
6.4.3 Algorithm Modifications for Workload- and Scheduling Preference-based Constraints 165
6.5 Experimental Design 167
6.5.1 Materials and Setup 168
6.5.2 Procedure 168
6.6 Results 171
6.6.1 Participants 171
6.6.2 Results for Situational Awareness 172
6.6.3 Results for Scheduling Preferences 175
6.6.4 Results of Varying Workload 178
6.7 Discussion 179
6.7.1 Design Guidance for Roboticists 179
6.7.2 Limitations and Future Work 180
6.8 Conclusions 184

7 Contribution and Future Work **187**
7.1 Thesis Contribution 187
7.2 Recommended Future Work 189
7.2.1 Learning to Teach 189
7.2.2 Learning with Similarities and Differences 190
7.2.3 Robots that Support Rather than Degrade SA 191
7.2.4 Long-term Interactions 192
7.2.5 Further Areas of Consideration 193

List of Figures

1-1 This figure depicts human and robots working together in final assembly operations of the Boeing 777. Image is credited to The Boeing Company. 20

1-2 This figure depicts (left) an entire team of human operators required to operate (right) the MQ-9 Reaper. Images are credited to the United States Air Force. 21

1-3 This figure depicts (left) hospital service robots transporting meals within a hospital. These robots, deployed at Southmead Hospital (Bristol, England), are the Transcar LTC 2. They are developed by Swisslog Telelift LTC, a member of KUKA Group. To the right, a surgeon is depicted working with Penelope, a robotic surgical assistant developed by Robotic Systems and Technologies, Inc. (New York, NY). Penelope must be commanded by the surgeon via voice commands. Images are credited to KUKA and Robotic Systems and Technologies, Inc., respectively. 22

1-4 This figure depicts an outline of my thesis. 25

3-1 The tactical action officer aboard a naval vessel must coordinate a heterogeneous set of soft- and hard-kill weapons to defeat various enemy, anti-ship missiles. 82

3-2 A resource nurse must assimilate a large variety and volume of information to effectively reason about resource management for patient care. 90

3-3 Pseudocode for the Mock Heuristic. 100
3-4 Screen capture of the simulation used in to capture data for the nursing data set. 102
3-5 Sensitivity of machine learning techniques using the pairwise, pointwise, and naïve approaches. 104
3-6 Specificity of machine learning techniques using the pairwise, pointwise, and naïve approaches. 104
3-7 Sensitivity for a pairwise decision tree varying the number and proportion of correct demonstrations. 105
3-8 Specificity for a pairwise decision tree varying the number and proportion of correct demonstrations. 105
3-9 Sensitivity for a pairwise decision tree, tuned for leafiness, varying the number and proportion of correct demonstrations. 108
3-10 Specificity for a pairwise decision tree, tuned for leafiness, varying the number and proportion of correct demonstrations with homogeneous agents. 108
3-11 Sensitivity for a pairwise decision tree, tuned for leafiness, varying the number and proportion of correct demonstrations. The corresponding data set was comprised of schedules with heterogeneous agents. . . . 109
3-12 Specificity for a pairwise decision tree, tuned for leafiness, varying the number and proportion of correct demonstrations. The corresponding data set was comprised of schedules with heterogeneous agents. . . . 109

4-1 The COVAS architecture. 117
4-2 The total computation time for COVAS, as well as the time COVAS required to identify a solution superior to that resulting from a human expert's demonstration. 121
4-3 The total computation time needed for COVAS and the MILP benchmark to identify the optimal solution for the tested scenarios. 123

5-1 Experiment participant pictured receiving advice from the robotic decision support. 130
5-2 Images of the robot system in action on the labor floor. 140
5-3 This figure depicts the service robot's system architecture. 141
5-4 A resource nurse must assimilate a large variety and volume of information to effectively reason about resource management for patient care. 141
5-5 This figure depicts the cell extraction process. 142

6-1 Tercio takes as input a temporal constraint problem and finds a satisficing, flexible schedule by utilizing an analytical schedulability test to ensure a feasible solution. 161
6-2 This figure depicts a diagram of the laboratory room where the experiment took place. There were two locations where the human and robot workers could inspect part kits during a fetching task, and two locations where the human workers built part kits. 167
6-3 This figure depicts participants' average SA scores for Questions 1, 5, 9, and 13 in the post-trial questionnaire shown in Table 6.1, as a function of the degree of automation over scheduling decisions. The standard error of the mean is shown as whisker bars. Note that a lower score indicates a better situational awareness. 173
6-4 This figure depicts the average of the medians of participants' responses to Likert-response Questions 2-4, 6-8, 10-12, and 14-16 under the autonomous, semi-autonomous and manual conditions. The standard error of the mean is shown as whisker bars. 174
6-5 This figure depicts the mean and standard error of the amount of work, in seconds, assigned to the participant by the robotic teammate. Horizontal bars with an asterisk denote statistical significance ($p < 0.01$). 177

Chapter 1

Introduction

Robotic systems are on the verge of revolutionizing almost every facet of our jobs. In manufacturing, robots are being utilized not only to fetch parts but also to directly contribute to value-added tasks such as drilling and fastening of aerospace structures (Figure 1-1). In military operations, remotely-piloted aerial systems (RPAS), such as the MQ-9 Reaper shown in Figure 1-2, are enabling operators to safely, and more precisely, conduct missions in domains too dangerous for human pilots. In healthcare, service robots are becoming increasingly utilized across a wide range of clinical settings. The aim of these robots is to reduce the burden on healthcare professionals by transporting supplies between care centers (Figure 1-3).

One of the most critical challenges for robots in these domains is to improve the ability of humans to operate more efficiently and safely given a finite set of resources (e.g., manpower, time, and money). However, resource optimization is one of the most costly and challenging aspects across almost every sector in the economy – with or without robots. For example, in healthcare, poor systems design and inefficient scheduling of resources can have drastic consequences on patient wait times. Patients with non-urgent needs who experience prolonged wait times have higher rates of non-compliance and missed appointments [129, 203]. Prolonged wait times and other inefficiencies in patient care contribute to the dissatisfaction and burnout of healthcare providers [229]. Recently, The United States Office of the Inspector General investigated allegations of gross mismanagement of resources at hospitals managed

Figure 1-1: This figure depicts human and robots working together in final assembly operations of the Boeing 777. Image is credited to The Boeing Company.

by the Veterans Health Administration and released a report finding that "significant delays in access to care negatively impacted the quality of care." [28] This issue is so critical that the Institute of Medicine recently released a report highlighting the need for better practices in scheduling and resource optimization [28]. However, healthcare is not the only industry that needs effective scheduling and resource optimization.

In automotive manufacturing, BMW produces approximately 1 car every 60 seconds in its facility in Spartanburg, South Carolina. In this plant, approximately 80%[1] of the cars are customized for individual customers. This customization requires the tight choreography of supply chain management and assembly. When a part shortage for one car on an assembly line occurs, every car is held until the conflict is resolved. Every 60 seconds spent re-scheduling work in response to a disturbance costs the company tens of thousands of dollars[2]. The Boeing Company similarly offers a high level of customization to its customers. Building an airplane is a complex process. For example, construction of a Boeing 747 requires the assembly of 6 million individual parts, a subset of which are customized for each patron. Every minute re-scheduling in response to dynamic disruptions in the build process can cost in excess of $\$100,000$[3].

[1]Statistic available from BMW Group Plant Spartanburg via `https://www.bmwusfactory.com/manufacturing/production-overview/`

[2]BMW's facility in Spartanburg, SC produced $36,580$ cars in March 2015. The base price of the cheapest car produced at the facility is $\approx \$38,500$. Assuming a 24/7 work week, that results in $\approx \$31,548$ of revenue earned/lost every minute. Numbers are courtesy of BMW USA.

[3]Boeing's facility in Renton, WA is increasing production of Boeing 737 aircraft to 47 per month by 2017. RyanAir and Boeing recently agreed to a purchase of 100 Boeing 737-Max aircraft for \$11 billion, which is approximately \$110 million per plane. Assuming a 24/7 work week, that results in $\approx \$108,603$ of revenue earned/lost every minute. Numbers are courtesy of The Boeing Company.

Figure 1-2: This figure depicts (left) an entire team of human operators required to operate (right) the MQ-9 Reaper. Images are credited to the United States Air Force.

The military is also highly invested in the effective use of resources. In naval conflict, defending one's ship from enemy anti-ship missiles is a challenging task. This problem requires the weighing of the relative benefits and detriments of using interceptor missiles to attempt to destroy the incoming anti-ship missile engagement versus deploying decoys to attempt to divert the attack. Interceptors are relatively easy to schedule: it is merely a one-to-one matching problem. The downside, however, is that it costs two orders of magnitude[4] more to manufacture an interceptor than it costs the enemy to build the missile. On the other hand, decoys and countermeasures are a substantially cheaper alternative to interceptor missiles. Yet, effectively deploying these decoys and countermeasures is challenging. A single decoy can affect multiple incoming missiles and do so in different ways. Further, defeating a single missile can require multiple decoys. The navy currently does not have a set doctrine for deploying countermeasures, and these countermeasures (e.g., a swarm of UAVs) lack their own ability to reason about how to best deploy to protect the ship. Instead, the navy relies on the expertise of naval tactical action officers to act.

These challenges in effectively utilizing resources are pervasive across many domains and are inherent to the problem of determining which workers should complete which tasks, when, where, and how. To enhance the productivity and safety of the

[4] The Israel-US developed Iron Dome system launches interceptors, which cost $\$50,000$. The system was developed to defeat Hamas-fired rockets, which cost only $\$500 - 1,000$ according to [173].

Figure 1-3: This figure depicts (left) hospital service robots transporting meals within a hospital. These robots, deployed at Southmead Hospital (Bristol, England), are the Transcar LTC 2. They are developed by Swisslog Telelift LTC, a member of KUKA Group. To the right, a surgeon is depicted working with Penelope, a robotic surgical assistant developed by Robotic Systems and Technologies, Inc. (New York, NY). Penelope must be commanded by the surgeon via voice commands. Images are credited to KUKA and Robotic Systems and Technologies, Inc., respectively.

environments in which humans work, researchers and industry practitioners alike have sought to use the power of robots to improve productivity and decrease the danger of tasks humans currently perform. However, the current method of deploying these systems is fundamentally limited and prevents us from realizing their full potential. The key problem is that robots are told explicitly what to do, when to do it, and how to do it. These systems do are being deployed without the ability to autonomously reason about the coordination their teams.

For example, in healthcare, service robots have been introduced to transport food and supplies between care centers in hospitals, such as those shown in Figure 1-3. Rather than the service robots being able to anticipate when and where extra supplies might be needed, nurses and doctors must take time away from patients to program the robots' tasks when the need arises.

Final assembly of automotive and aerospace assemblies has long been the domain of human workers. These workers must complete arduous tasks that place undue strain on the human body. Robots, such as those pictured at Boeing in Figure 1-1, have been introduced to offset this physical burden. However, whenever a part is late, a manufacturing error occurs, or a customer demands a change, the robots lack the ability to automatically adapt to the needs of the environment. Rather, humans must reprogram and reschedule the robotic work. The time required to manually reprogram these robots is expensive and erodes the benefits and feasibility of their deployment.

For naval fleet defense, the U.S. Navy is developing robotic decoys and countermeasures (e.g., drones), that can be repositioned dynamically to respond to a barrage of enemy anti-ship missiles. However, these robots lack the understanding of their environment to automatically adapt to that environment. Instead, naval operators must reposition their ships and the countermeasures when a threat arises. In operating even a single unmanned asset (Figure 1-2), an entire team of human operators is required. Even seconds taken to plan a new configuration and communicate that configuration to the entire team of human operators can leave too little time to execute the plan before the enemy missiles impact the fleet.

Rather than a paradigm where one or more human professionals are required to supervise a single robotic system, I see a future where we can flip this ratio on its head – a future where a single human operator can work among an entire team of support robots. For this transformation to happen, we need to give robotic systems the ability to learn to operate as independent contributors to human-robot teams. Surgical assistants, such as Penelope, should learn by demonstration and experience, just as human scrub technicians would, which instruments the surgeon needs *before* that surgeon requests them. Planetary rovers should be able to learn to identify scientific objects and how to carry out the logistics of the mission, just as a human astronaut would. RPAs should infer the needs of their team, identifying targets of interest, and plotting routes to maximize mission efficiency.

The challenge for robots then must be to learn how to coordinate their activities with human team members, just as a human apprentice would. However, this learning process is not a simple task. At its core, scheduling is a computationally complex problem that involves deciding which members of the team should complete which jobs, when, where, and how. Scheduling is an important challenge across problem domains - even without robotic assets. Further, while there are human domain experts who solve these logistics problems with amazing efficiency, these experts are not readily able to document or codify their knowledge into a *policy* for a robot to execute [45, 206]. What experts can provide are the important features of a problem – what they reason about – such as the deadline of a task, where it should happen, and how far away the necessary resources are [45, 206]. Thus, we need to give robots the ability to take as input the key features of how human experts reason about task completion, coupled with demonstrations by those experts, to be able to learn from demonstration just as a human apprentice does.

The challenge I pose in my thesis is to give a robotic system the ability to learn the strategies employed by domain experts and scale beyond a "single-expert, single-apprentice" model. By giving a single robot the ability to learn as an apprentice, that robot could share its knowledge with teams of robots and accomplish more than a single human apprentice. Further, these robotic systems could finally act as true, collaborative teammates rather than drones that must be micromanaged.

There are three key challenges I identified in my thesis that must be considered to realize my vision for a robotic apprentice. First, robots must be able to learn the rules-of-thumb and heuristics that human domain experts use to coordinate the activities of their teams. Second, robots must be able to transcend the power of the demonstrator. Expert human teams do perform an efficient choreography to accomplish joint missions, such as in search and rescue. However, humans are not perfect nor are their demonstrations. Robots should be able to leverage the value of the demonstrations they do receive, but they should also be able to reason about the structure of the problem (i.e., the goal and constraints) and use the power of computation to provide solutions better than the human expert could.

Finally, these systems must be able to contribute to their human counterparts – not just in theory but in practice. This practice entails considering the human factors aspects of human-robot teaming. For example, embodied intelligence and the anthropomorphism of the robotic system has been shown to alter the level to

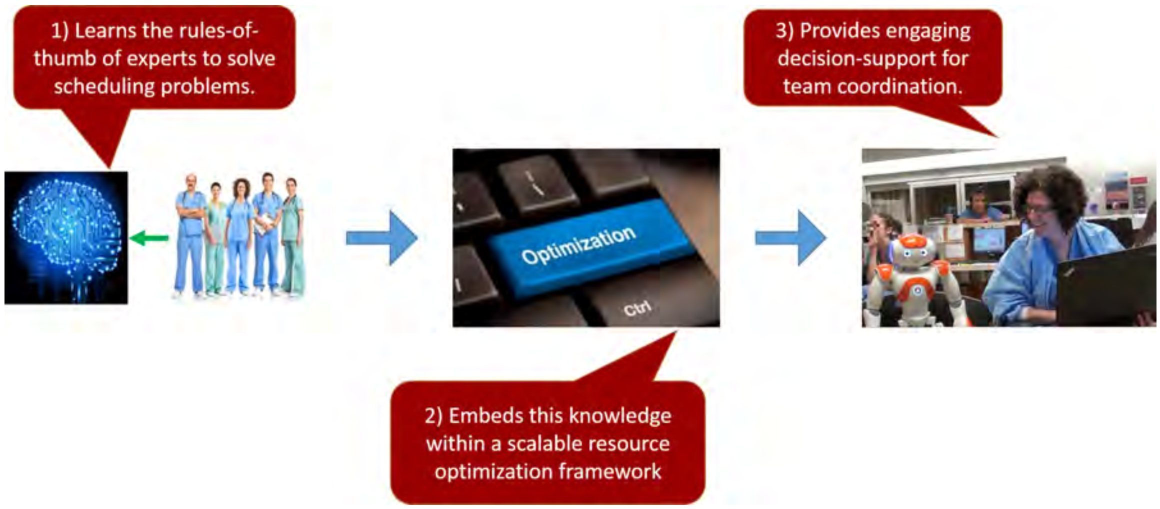

Figure 1-4: This figure depicts an outline of my thesis.

which humans trust and rely on the system [215]. Garnering too much trust in a faulty system would have dangerous consequences [64]. Further, there are important considerations for operator workload, situational awareness, and workflow preferences, which all can harm team fluency if the robotic system is designed in isolation of these effects. As such, I conduct and present in this thesis a set of human subject experiments studying the effects of embodiment, workload, situational awareness, and workflow preferences to provide design guidelines for deploying apprentice service robots.

A graphical depiction of my thesis is shown in Figure 1-4.In my thesis, I develop a new machine learning-optimization paradigm to enable service robots to learn to become supportive members of a human-robot team. The aims of my thesis are to develop an autonomous system that 1) learns the heuristics and implicit rules-of-thumb developed by human domain experts from years of experience, 2) embeds and leverages this domain knowledge within a scalable resource optimization framework to reduce the computational search space, and 3) is designed to provide decision support in a way that engages users and benefits them in their decision-making process. By intelligently leveraging the ability of humans to learn heuristics and the speed of modern computation, we can begin to solve the challenging requirements of real-time decision-support in healthcare, military operations, and manufacturing.

This chapter serves as an executive summary of the innovations and findings in this thesis. The following sections mirror the structure of the thesis. In Section 1.1, I formulate a novel machine learning technique, apprenticeship scheduling, to learn the rules-of-thumb and heuristics used by human domain experts to efficiently solve scheduling problems. I validate this technique on both a synthetic and a real-world data set by showing the apprenticeship scheduler is able to learn a high-quality model for scheduling based on demonstration. In Section 1.2, I extend my apprenticeship scheduling technique to encompass an entire machine learning-optimization framework. This framework, COVAS, is able to leverage the power of good, but imperfect human demonstration on relatively small problems to more efficiently find optimal solutions for larger problems. Finally, in Section 1.3, I give life to my apprenticeship scheduling algorithm, placing it in an embodied robotic system in a hospital, where the robot learns from expert nurses how to coordinate patients and their care staff. Because of the danger of humans potentially over-relying on the system and not performing their own due diligence to ensure the system is making valid decisions, I conduct a user study assessing the risk. I find that a robotic system garners more appropriate levels of reliance, reducing errors made by the human-robot team in comparison to a computer-based system.

1.1 Apprenticeship Scheduling: Learning to Schedule from Human Experts

In this chapter, I propose a new computational technique capable of learning the rules-of-thumb and heuristics domain experts employ to efficiently solve complex scheduling problems. To learn these rules-of-thumb and heuristics, I develop an apprenticeship learning technique, which is specifically suited for scheduling. The key to my approach is the use of pairwise comparisons between the action taken (e.g., schedule an agent to complete a task at a given time) and each action not taken (e.g., unscheduled tasks during that same time), at each moment of time, to learn the relevant model parameters and scheduling policies demonstrated by the training examples. I validate my approach using both a synthetic data set of solutions for a variety of scheduling problems and two real-world domains. The first is a real-world dataset of demonstrations from human experts solving a variant of the weapon-to-target assignment problem [150]. The second is a real-world dataset of resource nurses (i.e., charge nurses) in a hospital deciding how to assign care staff for patients and where to move those patients during their time in the hospital. The synthetic and real-world problem domains I use to empirically validate my approach represent two of the most challenging classes within the taxonomy of scheduling problems established by Korsah et al. [139].

Goal:

The problem of optimal task allocation and sequencing with upper- and lowerbound temporal constraints (i.e., deadlines and wait constraints) is NP-Hard [22], and real-world scheduling problems quickly become computationally intractable. However, human domain experts are able to learn from experience to develop strategies, heuristics, and rules-of-thumb to effectively respond to these problems. The challenge I pose is to autonomously learn the strategies employed by these domain experts; this knowledge can be applied and disseminated more efficiently with such a model than with a "single-expert, single-apprentice" model.

In this chapter, I develop a technique, which I call "apprenticeship scheduling," to capture this domain knowledge in the form of a scheduling policy. The objective is to learn scheduling policies through expert demonstration and validate that schedules produced by these policies are comparable in quality to those generated by human or synthetic experts. My approach efficiently utilizes domain-expert demonstrations without the need to train within an environment emulator. Rather than explicitly modeling a reward function and relying upon dynamic programming or constraint solvers, which become computationally intractable for large-scale problems of interest, my objective is to use action-driven learning (i.e., learning a function that maps states to actions) to extract the strategies of domain experts to efficiently schedule tasks.

Approach:

My apprenticeship learning algorithm is specifically suited for scheduling. The approach I use is inspired by work in webpage ranking [120, 194]. The web is often modeled as a graph, where webpages are nodes, and links between those webpages are directed arcs. Similarly, scheduling problems are often modeled as graphs with nodes to represent the start and finish events of individual tasks as well as directed arcs describing wait and deadline constraints that affect task events. The commonality of these models provides a suitable analogy for capturing the complex temporal dependencies (i.e., precedence, wait, and deadline constraints) relating tasks within a scheduling problem.

Within webpage ranking, there are two fundamental approaches to ranking: pointwise and pairwise. Pointwise ranking involves learning to predict the relevance of an individual webpage given a vector of features describing that individual webpage. In pairwise ranking, a model is learned to predict whether one webpage is more important than another by comparing (e.g., subtracting) the feature vectors describing each webpage.

The key to my approach is using a hybrid pointwise-pairwise ranking model to capture the knowledge demonstrated by the expert. For scheduling, I perform pairwise comparisons between the features of scheduling actions taken (e.g., schedule agent a to complete task τ_i at time t) and the set of actions not taken (e.g., unscheduled tasks at time t) to learn the relevant model parameters and scheduling policies demonstrated by the training examples.

However, by only using pairwise comparisons, one loses the *context*, or high level features that describe the overall task set. For example, information about the proportion of workers who are busy versus idle may affect the decision a human domain expert makes. Yet, if this proportion is encoded as a feature in a pairwise comparison, the result is uninformative (i.e., their difference is zero) because both will have the same value . Thus, my approach preserves high level, contextual information as pointwise features.

With pointwise terms describing the context of the state of the scheduling problem, and the pairwise comparisons made between the action taken and the corresponding set of actions not taken (and vice versa), I train a classifier to model how the expert chooses to take one action but not another. I show that this hybrid pointwise-pairwise approach is able to capture $> 95\%$ of the knowledge provided by a sophisticated scheduling demonstrator.

Results and Contributions:

I validated my apprentice scheduling approach on three data sets: a synthetic data set for studying a variant of the multiple-vehicle routing problem, a real-world dataset for studying a variant of the weapon-to-target assignment problem [150], and a second real-world dataset for studying the role of a resource nurse in a hospital's labor and delivery floor. By showing the ability of the apprenticeship scheduler to function across multiple data sets, I build support for the wide applicability of my approach.

For the first investigation, I studied a synthetic variant of the multiple-vehicle routing problem in which the goal was to find the shortest possible route for a set of vehicles to pick up and deliver packages given a set of temporal ordering constraints (e.g., one location must be visited before another), time windows (e.g., a location must be visited after 11:00 am but before 2:00 pm), and physical constraints (e.g., no two vehicles can be at the same place at the same time). I constructed a mock expert that applies one of a set of scheduling rules based on individual problem characteristics. The challenge for the apprenticeship scheduler was then to learn a model for how the mock heuristic acts based upon a set of schedules created by the mock expert. I found that the apprenticeship scheduler, using the pairwise approach and a decision tree classifier, works effectively at learning from the mock expert. Specifically, with only 15 demonstrations from a noisy expert (i.e., one that makes mistakes 20% of the time), the apprenticeship scheduler matched the decision-making of the mock expert 60% of the time. As the number of demonstrations increased, and the noisiness of the demonstrator decreased, this accuracy increased to $\sim 95\%$.

Second, I studied a real-world data set with actual human domain experts. The problem domain was anti-ship missile defense (ASMD), which is a complex variant of the weapon-to-target assignment problem. In this problem, a human naval officer is faced with an adversary who launches a set of anti-ship missiles. The officer must decide how to deploy a set of decoys and countermeasures to defeat the anti-ship missiles thus defending his/her ship. Specifically, the officer must decide which decoy(s) to deploy, as well as when and where to deploy them, to defeat the intended missiles. The dynamics of the missiles and decoys are complex, and the inaccurate deployment of a decoy could cause a missile to impact a ship when it might have missed the ship had the deployment not been made. I collected a dataset of human operators performing ASMD in a military training simulation. I trained the apprenticeship scheduler and it achieved a statistically significant improvement in performance over the average human demonstration.

Third, I evaluated the efficacy of apprenticeship scheduling with actual human domain experts in health care. The problem domain was patient care in labor and delivery in an obstetrics and gynecology ward at Beth Israel Deaconess Medical Center (BIDMC). Here, a single nurse, the resource nurse, must decide which patients should go to which rooms at which times, and ensure the right nurses and physicians are there to care for them at the right time. The resource nurse essentially runs "air traffic control" operations for patient care. I collected a data set of expert resource nurses managing patients in a simulation of the labor floor. I trained my apprenticeship scheduling algorithm on this data set and asked nurses and physicians to evaluate the quality of advice given by the algorithm. These healthcare professionals affirmed the advice 90% of the time.

By demonstrating that the apprenticeship scheduler can imitate both sequential decisions and the quality of the overall schedule, I showed that the apprenticeship scheduling framework is a viable technique for learning to schedule from human experts. The technique withstands imperfect demonstrations and can efficiently learn from relatively small data sets.

1.2 Learning to Make Super-Human Scheduling Decisions

Imitation learning for scheduling is critical to solving scheduling problems. Simply knowing the objective function or reward signal is insufficient given the computational complexity of solving for the optimal schedule. Rather, one needs a means (i.e., a policy) of creating a good schedule. While my apprenticeship scheduling algorithm is quite successful in imitating experts and creating good solutions, it is limited in two ways. First, if the human experts provide poor demonstrations, the learned policy will be similarly poor. Second, the policy lacks a feedback mechanism (i.e., a reward signal or objective function) to guide it toward the intended, optimal schedule.

In this chapter, I build off my initial apprenticeship scheduling work to develop an optimal scheduling framework, which I call Collaborative Optimization via Apprenticeship Scheduling (COVAS). COVAS bridges the fields of machine learning and optimization to learn how to efficiently and optimally solve scheduling problems. My enhanced approach withstands imperfect human demonstrations, and leverages knowledge of a global objective function to provide globally optimal solutions. I demonstrate that COVAS both produces globally optimal solutions at a rate 9.5 times as fast as an optimization approach that does not incorporate human expert demonstration.

Goal:

Recent research has aimed to capture goal-based knowledge obtained through demonstration via a process known as reward learning [2, 19, 115, 137, 269, 189, 247, 248, 252, 270]. These techniques are typically comprised of two parts. First, regression is used to infer a reward or objective function. Second, a solution is generated to maximize that reward function.

One common technique for reward learning (RL) is inverse reinforcement learning (IRL). In IRL, the second step involves constructing a policy via reinforcement learning, which requires state space enumeration and exploration [2, 115, 137, 269, 189, 247, 248, 252, 270]. However, as noted in prior works [89, 262, 255, 268], the large amount of data required to regress over the large state spaces associated with scheduling problems remains daunting, and RL-based scheduling solutions exist only for simple problems [262, 255, 268].

There is a body of work by Berry et al., that developed a reward learning approach, called PTIME, for scheduling problems [19]. PTIME solicits the reward function via questionnaire and generates the optimal solution by solving an integer linear program. However, the computational complexity of this approach is exponential, and is, therefore, limited to small problems that can be efficiently solved by an integer linear program [51].

Solving for an objective function's optimal solution through search is computationally complex and limits the algorithm's scalability. This challenge is particularly true for combinatorial optimization problems, such as scheduling. This challenge motivated me to develop apprenticeship scheduling, which learns a mapping from states to actions. With this mapping, apprenticeship scheduling can construct an empirically good schedule (i.e., one with a high, if suboptimal, objective function score) quickly, in polynomial time. However, sequentially evaluating the policy can result in small errors that result in large, cumulative deviations from the optimal sequence. Thus, policy learning must be combined with a feedback mechanism to correct these deviations.

In this chapter, I extend my apprenticeship scheduling method to a collaborative optimization via apprenticeship scheduling (COVAS) technique to provide globally optimal scheduling solutions. COVAS is comprised of two components: apprenticeship scheduling and optimization. COVAS is initialized by a policy learning phase to learn from human demonstration and then uses the resulting policy to generate an initial seed solution to a mathematical optimization. This seed provides a tight bound (validated empirically in Chapter 3) on the value of the optimal solution, which can be used during the branch-and-bound search to prune large swaths of the search tree. Second, the seed solution can be improved through local search to find successively better solutions. I show this policy can be used in conjunction with a MILP solver to substantially improve computation time in solving for the optimal schedule.

My work is distinguished from prior works that incorporated policy gradient descent or variants of q-learning in that COVAS is guaranteed to produce a globally optimal solution to the scheduling problem. Also, COVAS can be employed as an anytime algorithm that provides a bound on the sub-optimality of the solution. To my knowledge, my work is the first to develop and demonstrate an approach to learning through human demonstrations to efficiently produce optimal solutions for complex real-world scheduling problems.

Approach:

COVAS takes as input a set of domain expert scheduling demonstrations (e.g., Gantt charts) that contains information describing which agents complete which tasks, when, and where. These demonstrations are then passed to an apprenticeship scheduling algorithm that learns a classifier, $f_{priority}(\tau_i, \tau_j)$, to predict whether the demonstrator(s) would have chosen scheduling action τ_i over action $\tau_j \in \boldsymbol{\tau}$.

Next, COVAS uses $f_{priority}(\tau_i, \tau_j)$ to construct a schedule for a new problem. COVAS creates an event-based simulation of this new problem and runs the simulation in time until all tasks have been completed. To complete tasks, COVAS uses $f_{priority}(\tau_i, \tau_j)$ at each moment in time to select the best scheduling action to take. I describe this process in detail in the next section.

COVAS provides this output as an initial seed solution to an optimization subroutine (i.e., a MILP solver). The initial solution produced by the apprenticeship scheduler improves the efficiency of a search by providing a bound on the objective function value of the optimal schedule. This bound is used to inform a branch-and-bound search over the integer variables [22], enabling the search algorithm to prune areas of the search tree and focus its search on areas that can yield the optimal solution. After the algorithm has identified an upper- and lowerbound within some threshold, COVAS returns the solutions that have been proven optimal within that threshold. Thus, an operator can use COVAS as an anytime algorithm and terminate the optimization upon finding a solution that is acceptable within a provable bound.

Results and Contributions

First, I trained COVAS' apprenticeship scheduling algorithm on demonstrations of experts' solutions to unique ASMD scenarios (except for one "hold-out" scenario). I then tested COVAS on this hold-out scenario. In addition, I applied a pure MILP benchmark on this scenario and compared the performance of COVAS to the benchmark. I generated one data point for each unique demonstrated scenario (i.e., leave-one-out cross validation) to validate the benefits of COVAS. I show that COVAS is not only able to improve overall computation time for the optimal solution, but it also substantially improves computation time for solutions that are superior to those produced by human experts. The average improvement in computation time with COVAS is 6.7x and 3.1x, respectively. Second, I evaluated COVAS' ability to transfer prior learning to more challenging task sets. I trained COVAS on a level in the ASMD game in which a total of 10 missiles of varying types came from specific bearings at given times. I randomly generated a set of scenarios involving 15 and 20 missiles, with bearings and times randomly sampled with replication from the set of bearings used in the 10-missile scenario. I found that the average improvement in computation time with COVAS was 4.6x, 7.9x, and 9.5x, respectively. This evaluation demonstrates that COVAS is able to efficiently leverage the solutions of human domain experts to quickly solve problems twice as large as those the demonstrator provided for training.

1.3 Robot Embodiment as a Scheduling Apprentice

Thus far, I have discussed the need for and benefits of learning from human demonstration how to solve complex scheduling problems. I first formulated a novel machine learning technique to capture the rules-of-thumb and heuristics of human domain experts. Next, I extended this technique to encompass an entire machine learning-optimization framework. This framework, COVAS, can use good, but imperfect, human demonstrations on smaller scheduling problems to more efficiently, as well as optimally, solve larger problems. Thus, COVAS is able to scale beyond the power of the expert by leveraging the knowledge demonstrated by that expert. However, until this point, I have only solved a problem in simulation – I have yet to demonstrate apprenticeship scheduling embodied in a physical platform.

In this chapter, I give life to apprenticeship scheduling by fielding and testing the efficacy of a robot working alongside human experts in a life and death medical domain: the labor and delivery floor in the obstetrics and gynecology department at BIDMC. In hospitals, service robots are starting to take hold as key tools in carrying out patient care, for example, by transporting food and medicine. However, these systems must be explicitly tasked and supervised. Instead, I envision a hospital where robots learn to anticipate the needs of their human counterparts and adapt to their dynamic environment, just as a human nurse would. I believe apprenticeship scheduling has the potential to give robots the ability to learn to anticipate their teammates' needs and seamlessly integrate into the healthcare environment.

I take steps toward answering two important questions in realizing this vision. First, I ask, "Are robots the right way to improve resource optimization in labor and delivery?" Using apprenticeship scheduling, I can give a robot the ability to participate in physical and cognitive labor and delivery tasks; however, is a robot the right form factor for this assistance? Anthropomorphism of the robotic system has been shown to alter the level to which humans trust and rely on the system [215]. I conduct a novel human-subject experiment to investigate embodiment in human-robot interaction (HRI) in a labor and delivery unit and find that a robotic system, in fact, garners more appropriate trust and reliance than a non-embodied system. Second, given this support, I ask, "How can robots improve resource optimization in labor and delivery?" To answer this question, I discuss the development and deployment of an entire robotic system that can autonomously infer the state of the labor floor, make effective decisions, and verbally communicate with nurses and doctors to assist in patient care.

Goal:

Service robots are being increasingly utilized across a wide spectrum of clinical settings. They are deployed to improve operational efficiency by delivering and preparing supplies, materials, and medications [24, 63, 111, 178, 191]. However, these robots are not yet well-integrated into the healthcare delivery process – they do not operate with an understanding of patient status and needs and must be explicitly tasked and scheduled. Instead, I propose embedding these service robots and others like them with the ability to anticipate the needs of the hospital and act with some autonomy.

In Chapter 3, I demonstrated that an apprenticeship scheduling algorithm could in fact learn from nurses responsible for managing resources in a hospital. However, there are two questions that must be answered before such a system can be realized. First, is a robot the correct form factor for providing assistance in labor and delivery? Second, how do we design an entire robotic system to embody apprenticeship scheduling in a robotic assistant?

Approach:

While the overarching goal is to support human professionals in the field, it is important to first ensure we know the right way to deliver that support. Given the concerns of how embodiment can affect the trust and reliance of human operators, I conducted a user study with one physician and sixteen registered nurses.

In the experiment, two independent variables were manipulated. The first independent variable was the embodiment of the apprenticeship scheduler: either a robot or a computer-based decision support system would use the apprenticeship scheduler to offer advice. The second variable was the quality of that advice: the decision support system would sample high or low-quality advice from the apprenticeship scheduler. These two independent variables, each with two factor levels, result in a 2x2 experimental design with four total conditions. I employed a within-subjects experimental design, meaning that participants experienced all four conditions.

During a given trial, participants would play a simulation of a shift on a labor and delivery floor playing the role of the resource nurse. During the simulation, patients would arrive in the waiting room, and the decision support system would offer advice on where to place the patient and who could care for that patient. The participant would then either accept or reject the advice. At the end of each simulated shift, the participants would respond to a questionnaire to measure their subjective experiences working with the decision support system. The questionnaire was used to measure trust, while the individual accept/reject decisions were used to measure reliance on the system in each factor level for each independent variable.

Results and Contributions

This chapter presents two novel contributions to the fields of robotics and healthcare. First, through human subject experimentation with a physician and registered nurses, this is the first study to my knowledge to be conducted involving experts working with a robot on a real-world complex decision-making task comparing trust in and dependence on robotic versus computer-based decision support. Previous studies have focused on novice users and/or simple laboratory decision tasks [10, 58, 131, 154]. My findings provide the evidence that experts performing decision-making tasks may not be as susceptible to the negative effects of support embodiment as previously thought [10, 58, 131, 154]. Furthermore, embodiment yielded performance gains compared with computer-based support during periods following a change in the quality of advice from the decision-support system. This provides encouraging evidence that intelligent service robots can be safely integrated into the hospital setting.

Second, the first test demonstration of a robotic system was conducted in which the robot assisted resource nurses on a labor and delivery floor in a tertiary care center. The robot utilized machine learning computer vision techniques to read from a whiteboard the current status of the labor floor and make suggestions about resource allocation, and it also used speech recognition to receive feedback from the resource nurse. To my knowledge, this is the first investigation to field a robotic system that provided real advice, regarding real patients, in real time, in a hospital to aid in the coordination of resources required for patient care.

1.4 Situational Awareness, Workload, and Workflow Preferences with Service Robots

Thus far, I have shown that apprenticeship scheduling can learn from human experts how to solve complex scheduling problems, use that knowledge to generate solutions better than the human expert, and even provide decision-support in an embodied platform. In Chapter 3, I investigated the effects of embodying apprenticeship scheduling in a service robot. I found that embodiment has a strong positive effect in improving human-robot team fluency versus a computer-based decision-support system. However, embodiment is but one of many concerns in human factors.

Introducing any form of automation into a human environment can be hazardous. For example, automation tends to degrade humans' situational awareness, impairing their ability to intervene in the event of a robot's failure [74, 75, 77, 125, 214]. Many fatal airplane crashes can be attributed to a pilot's loss of situational awareness while interacting with cockpit automation [179, 180, 181, 182, 183].

Workload assignment is another key issue in human factors [198, 233, 249, 257]. It has been shown in prior work that human performance is highly dependent upon workload [233, 249, 198, 257, 204]; further, workload that is too heavy or too light can degrade performance and contribute to a loss of situational awareness [249, 204].

Understanding and incorporating workflow preferences is also essential for safe, effective human-machine teaming [4, 104, 142, 144, 187]. In manufacturing, human teams can develop individualized workflow preferences that are not shared by other teams in the same environment; consequently, a member of one team may be unable to effectively replace a worker on another team without a period of adjustment.

This chapter presents a series of three human-subject experiments to investigate situational awareness, workload, and workflow preferences in human-robot teaming. In these experiments, a human participant works with a second human – a confederate – and an autonomous robot to complete a set of tasks in a mock manufacturing environment. While this study was not performed in situ, it was conducted in a highly controlled setting with high internal validity. Based on the findings of this work, I am able to develop design guidelines for roboticists deploying autonomous service robots that employ apprenticeship scheduling or other scheduling technologies.

Goal:

Human-robot teaming has the potential to increase the productivity of human labor and improve the ergonomics of manual tasks. Based on recent industry interest in fielding human-robot teams, researchers have been investigating how best to include a human in the decision-making loop as a member of a human-robot team [3, 8, 13, 52, 68, 92, 101, 109, 202, 221, 267]. However, the intricate choreography required to safely and efficiently coordinate human-robot teams represents a challenging computational problem. The choreography of a team in which one must allocate and sequence tasks with upper- and lowerbound temporal constraints is known to be NP-Hard [22].

Fully autonomous solutions to this problem have been recently proposed by both academic researchers [22, 91, 200] and industry practitioners [6]. While these new computational methods have significantly enhanced the ability of people and robots to work flexibly together, there has been little study into the ways in which human factors must influence the design of these computational techniques. Specifically, we must consider how situational awareness changes as a function of the level of robot initiative during the decision-making process, the consequences of varying the workload assigned by the robot to human agents, and how to include the workflow preferences of human team members into decision-making. Improvements assessed through simple measures of efficiency, such as task time, do not guarantee the long-term productivity and viability of the human-robot team.

This chapter proposes three novel human-subject experiments to investigate situational awareness, workload, and workflow preferences in human-robot teaming. The three hypotheses posit that increasing the robot team member's autonomy will degrade the participant's situational awareness. Further, having the robot assign too much or too little work to the participant, or disregarding the participant's workflow preferences, will decrease team fluency.

Approach:

The approach to evaluating these hypotheses consists of conducting three human-subject experiments to isolate the effects of automation on situational awareness, workload, and workflow preferences.

To study situational awareness, participants would work on the human-robot team to complete a set of manufacturing tasks. There were three conditions: manual, autonomous, and semi-autonomous. In the manual condition, the participant would allocate the tasks to each team member. In the autonomous condition, the robot would. In the semi-autonomous condition, the participant assigned tasks to himself, and the robot allocated the remaining tasks to itself and the second human teammate (i.e., the confederate). During the experiment, I would administer the SAGAT (Situation Awareness Global Assessment Technique) test [74]. For this test, all work by the team would cease, and the experimenter would ask participants to complete a questionnaire gauging the participant's situational awareness (e.g., "What task is the robot currently completing?"). These questions help ascertain whether the participant would have sufficiently knowledge in working memory to be able to re-schedule the team's activities manually should a disruption occur.

For studying workflow preferences, a human-subject experiment was conducted in which participants would work on the same human-robot team as in the situational awareness experiment. However, in this experiment, the robot is responsible for making all scheduling decisions. The experimenter asks participants which tasks they would prefer to complete from the total set of work assigned to the team. The robot would then take this information and schedule the team in one of three conditions. First, the robot would give the participants most of the tasks they prefer. In the second condition, the robot would give the participants most of the tasks they do not prefer. In the third condition, the robot ignores the preferences of the participants. By using subjective and objective measures of team fluency, one can gauge the effects of incorporating or ignoring participants' workflow preferences when working on a human-robot team.

Finally, to study workload, a third human-subject experiment was conducted assuming the same experimental setup as in the experiment studying workflow preferences. However, the robot now operates along two axes. First, the robot alters the amount of work assigned to the participant. Second, the robot either adheres to or goes against the workflow preferences of the human subject. These axes result in a total of four conditions. The effects of workload and workflow preferences on team fluency can be isolated by utilizing objective and subjective measures in each condition.

Results and Contributions

In the first study, situational awareness was found to vary as a function of the degree of autonomy a robotic agent has during scheduling. The human participants' awareness of their team's actions decreased as the degree of robot autonomy increased. This indicates the desire for increased autonomy and accompanying performance improvements must be balanced with the risk for – and cost resulting from – reduced situational awareness.

Team fluency was found to vary as a function of the workload given to a human team member by a robotic agent, and the manner in which a robot should include the workflow preferences of its human teammates in the decision-making process. Results of these studies indicate a complex relationship between preferences, utilization and the participants' perception of team efficiency. Participants prefer a robot to include his or her workflow preferences; however, participants demand that the robot also properly throttle the participants' workload.

Based on these findings, I provide a set of design guidelines to assist roboticists in developing service robots.

Chapter 2

Background

2.1 Introduction

My thesis is that we need to develop intelligent robotic assistants that have the ability to learn from human experts' demonstrations how to effectively integrate into the human workplace. This development requires not just a technical advance in learning from demonstration for team coordination, but also an understanding of the human factors consequences of integrating intelligent service robots into human-robot teams. This chapter lays the foundation for developing and translating techniques from artificial intelligence (AI) and human factors to develop robot apprentice schedulers. From this foundation, I develop a human-machine collaborative optimization technique that enables robots to learn from human demonstration strategies and rules-of-thumb for coordinating team resources (e.g., members of the team to complete a set of jobs). In later chapters, I translate this work to an embedded robot system to develop design guidelines for these systems. In addition to studying the effects of embodiment, I investigate situational awareness, workload, and workflow preferences of the human team members of these robotic systems.

To begin, I survey the state-of-the-art methods in scheduling of human-robot teams in Section 2.2. AI Planning and Scheduling is a broad area encompassing many related problems and techniques. Rather than survey the entire wealth of work in this field, I focus on techniques that are designed to efficiently coordinate teams of workers in time and space where they must negotiate temporal (i.e., deadlines and wait constraints) and spatial constraints (e.g., no two workers can be at the same place at the same time). The purpose of this review is to demonstrate a key point: As problems become more challenging due to the their size or constraint complexity, the solution techniques for those problems become more reliant on clever heuristics and problem decompositions.

Yet, manually developing these heuristics is challenging and not scalable. The ability of the researcher to translate these techniques across domains is fundamentally limited by the need to develop heuristics applicable to each domain. A core tenet of my thesis is that we can instead learn from human domain experts – practitioners – who efficiently solve these domain-specific problems on the fly. While these experts can provide the relevant features they use to reason about their scheduling problems, they are less able to codify a set of scheduling rules or a policy to encapsulate their knowledge [45, 206]. Thus, we need to develop machine learning from demonstration techniques especially suited for capturing the rules-of-thumb and heuristics of these domain experts for solving scheduling problems.

Learning from Demonstration (LfD), however, has not typically been applied to scheduling. In Section 2.3, I discuss the two primary techniques for LfD within the context of machine learning: goal and policy learning. Goal learning consists of inferring the intended goal of the demonstration (e.g., pour a cup of water), whereas policy learning entails learning to mimic the demonstration (e.g., how to pour). I discuss the relative merits of these approaches and initially motivate the need to first develop a policy learning approach for apprenticeship scheduling. Specifically, scheduling is a challenging NP-Hard problem, and simply knowing a goal does not allow one to compute a schedule to accomplish that goal. Rather, we need a mechanism to navigate toward a goal (i.e., a policy).

Policy learning, though, is only the first step in developing an intelligent service robot. I recommend a policy learning technique that can guide the construction of a team's schedule; knowing the goal is insufficient. Yet, at its worst, policy learning is dead-reckoning. While this need to dead-reckon is crucial to reducing the computational search space of scheduling problems, it is fundamentally limited. Specifically, without knowing the goal, we are unable to improve the solutions generated by the policy. Because human experts' demonstrations are good but imperfect, the learned policy will mirror this imperfection. My thesis seeks to blend machine learning and optimization to leverage imperfect demonstrations to provide optimal solutions. In Section 2.4, I present work in blending machine learning and optimization to generate solutions better than those found in the demonstration.

Once a new algorithmic technique is developed, one must next understand the consequences of embedding that algorithm in a physical system: a robot. This first challenge is, of course, to construct the system required to animate the robot – computer vision, natural language process, locomotion, etc. While these are interesting and important challenges, I investigate a second, more basic question: What are the human-factors consequences of placing a robot, designed to assist in team coordination (i.e., scheduling) into a human workspace? Do humans become amused[1] and complacent, no longer participating in the validation of the team's actions? What happens if that robot gives the human team members too much or little work to perform? What are the effects of a robot ignoring the workflow preferences of the human team members? This thesis is not the first to raise the concerns of embodiment, situational awareness, and workload: Section 2.5 serves as a review of this related work. However, this thesis makes two novel contributions within this body of work. First, embodiment of a robotic assistant has not been well-studied in the context of working with experts; prior work has typically focused on embodiment with non-expert populations (e.g., entertainment consumers). Second, situational awareness, workload, and workflow preferences have not been studied in the context of a robotic teammate capable of mixed-initiative scheduling. This thesis makes contributions to both of these areas.

[1]Literally, without thought.

2.2 AI: Scheduling

There is a wealth of prior work in task assignment and scheduling within the manufacturing field and other applications. Korsah et al. provided a comprehensive taxonomy [139] for the multi-robot task allocation and scheduling problem. The taxonomy describes the "degree of interrelatedness" amongst the robots' schedules, as shown below. Each successive category subsumes the complexity of the previous categories.

- No Dependences (ND) - Each robot's tasks can be performed at any time and without regard for other robots' activities.

- In-Schedule Dependencies (ID) - The order in which an individual robot performs its own tasks matters; however, that ordering is independent of the activities of other robots. The multiple-vehicle routing problem falls within this class.

- Cross-Schedule Dependences (XD) - The utility of one robot is directly affected by the scheduling commitment of another. Such dependencies occur when, for example, workers cannot be in the same physical location at the same time.

- Complex Dependences (CD) - The way in which a robot completes a task affects the way in which other robots can complete their own tasks. To illustrate this scenario, consider a problem in which bulldozers and firetrucks must work together in disaster response to put out fires in a city. The roads that bulldozers clear affect which routes firetrucks can take to put out the fires.

The problems I consider in this thesis include both XD and CD, which subsume the complexity of the ND and ID classes of complexity. What follows here is a review of current Artificial Intelligence (AI) planning and scheduling techniques designed to handle these types of problems, including MILP formulations, auction- and market-based methods and other approaches (e.g., hybrids of MILP and heuristic algorithms), and I discuss the applicability of these techniques to the problem at hand.

Nota bene: Korsah et al. state that, in many cases, the problem of task allocation with XD can be readily formulated and solved as a MILP [30, 152]. However, there is no standard formulation for CD problems. As such, solution techniques are almost exclusively heuristic in nature [139].

2.2.1 MILP/CP Solution Techniques

One of the most promising approaches to solving this class of problems has been the development of a hybrid algorithm incorporating MILP and constraint programming (CP) methods along with decomposition. Techniques based on Benders Decomposition [17, 85] are among the most widely used. Here, we briefly review this method using treatments by Hooker [107], Martin [168] and Christensen and Pedersen [50], and discuss its variants and applications.

Consider the linear program depicted in Equation 2.1. The objective of this program is to minimize the function $z = cx$, where x represents the decision variables, c the cost function coefficients and $\boldsymbol{A}$, b and a represent the constants or coefficients of the constraint equations.

$$\begin{aligned} \min\ z &= cx \\ \text{s.t.}\ \ \boldsymbol{A}x &\geq b \\ x &\geq a \end{aligned} \tag{2.1}$$

Benders Decomposition bifurcates the decision variables into x and y to reformulate the problem as shown in Equation 2.2. Here, solutions for x and y are constrained to their respective domains, D_x and D_y:

$$\begin{aligned} \min\ z &= cx + f(y) \\ \text{s.t.}\ \ \boldsymbol{A}x + g(y) &\geq b \\ x \in D_x, y &\in D_y \end{aligned} \tag{2.2}$$

To simplify the problem, a value $\bar{y}$ of y is chosen from D_y, as shown in Equation 2.3:

$$\begin{aligned} \min\ z &= cx + f(\bar{y}) \\ \text{s.t.}\ \ \boldsymbol{A}x &\geq b - g(\bar{y}) \\ x &\in D_x \end{aligned} \tag{2.3}$$

The dual of Equation 2.3 is then derived in Equation 2.4:

$$\begin{gathered} \max\ \beta \\ \text{s.t.}\ \ \boldsymbol{A}x \geq b - g(\bar{y}) \xrightarrow{x \in D_x, \bar{y} \in D_y} cx + f(\bar{y}) \geq \beta \end{gathered} \tag{2.4}$$

The optimal solution of β from Equation 2.4 provides a lowerbound for the value of the optimal solution in Equation 2.1. By carefully choosing values for $\bar{y}$, one can incrementally tighten the lowerbound on the optimal solution to Equation 2.1 and quickly cut the search space. Benders Decomposition works by constructing a cutting function, $z \geq \beta_{\bar{y}}(y)$, called the *Benders Cut*, that takes any value of $y \in D_y$ as input and returns the value for β that lowerbounds the optimal value for z from the primal problem (Equation 2.1). Different methods are necessary for deriving Benders Cut depending on whether y is linear [168, 17, 85] or integer-based (a.k.a. Logic-Based Benders) [107].

Various planning and scheduling applications have incorporated Benders Decomposition. Logic-Based Benders Decomposition in particular has been applied to solve job shop scheduling problems where n tasks must be optimally scheduled at m facilities [106, 108], where y represents the assignment of tasks to facilities, and x represents the sequencing of those tasks. Benders Cuts are generated by separating the allocation of tasks from the sequencing of those tasks. Reported results demonstrate optimal solutions for problems involving approximately 10 facilities and 50 tasks.

Cordeau et al. and Mercier et al. employed Benders Decomposition for the problem of aircraft routing and crew scheduling [54, 171]. These works heuristically decomposed aircraft routing from crew assignment and generated Benders Cuts for the resulting assignment sub-problem. They empirically demonstrated that this method produced more cost-efficient schedules than prior art. Rekik et al. similarly employed Benders Decomposition to schedule personnel shift-work [210]. They applied a hand-tailored set of *forward* and *backward constraints* to cut the search space and proved the correctness of these constraints. Rekik et al. showed that Benders Decomposition can be used to solve particularly challenging problems in which the *forward* and *backward constraints* do not sufficiently prune the search space.

While Benders Decomposition has served as the basis for many state-of-the-art scheduling algorithms, several alternative techniques have also successfully combined MILP and CP approaches. Jain and Grossmann [118] presented an iterative method that first solves a relaxed MILP formulation and then searches for the complete solution using CP. When applied to the scheduling of jobs for machines, the MILP relaxation solves for the assignment of jobs and the CP solves for the schedule. If a solution is identified, the algorithm returns the optimal solution; otherwise, the algorithm infers cuts based on the solution of the MILP relaxation and solves the new MILP relaxation using these cuts. Results were reported for problems involving up to 20 tasks and five machines. Li and Womer later improved on this work by employing a hybrid Benders Decomposition algorithm with MILP and CP solver subroutines [155], and reported that their method can solve problems involving 30 tasks and eight different agent types up to four times faster than a standard MILP, which would require hours. Ren and Tang [211] took a similar approach, but employed heuristic strategies to generate informative cuts in the event that the CP solver was unable to identify a feasible task sequence. A related method proposed by Harjunkoski et al. [95] utilized an iterative approach to producing task assignments and schedules. Ren and Tang, and Harjunkoski and Grossman empirically demonstrated that their algorithms can solve problems involving up to eight machines and 36 jobs; however, these works did not address problems with cross-schedule dependencies or task-task deadlines.

2.2.2 Auction, Market-Based, and Heuristic Solution Techniques

Auction methods and other market-based approaches to scheduling problems, such as those developed by Brunet et al., Choi et al. and Liu and Shell, also frequently rely on decomposition of the problem structure [31, 209, 160]. For example, Choi et al. developed the Consensus-Based Bundle Algorithm (CBBA), a decentralized task allocation method that uses a market-based approach where agents (i.e., unmanned aerial vehicles) bid for tasks [31, 209]. The objective function and constraints are decomposed with respect to agents, so that each agent can quickly solve for the value of its bid on each task. A subroutine then resolves inconsistencies that arise in agents' bids due to communication restrictions among agents. This technique treats the construction of each agent's schedule as independent of other agents' and precludes explicit coupling in each agent's contribution to the MILP objective function.

Many other techniques solve the task allocation problem efficiently [235, 160, 20, 21, 264, 16], but do not address the scheduling problem. For example, Sung et al. addressed the problem of task allocation within a multi-robot environment where each agent maintains a queue of tasks and partial information about other agents' queues [235]. During execution, agents communicate when possible and choose to exchange tasks using a heuristic approach. Sung et al. empirically demonstrated that their algorithm can solve problems involving up to six agents and up to 250 tasks but the problems did not involve cross-schedule dependencies or task deadlines.

Liu and Shell recently proposed a novel distributed method for task allocation via strategic pricing [160]. Their work builds upon prior approaches to distributed auction algorithms [20, 21, 264], runs in polynomial time and produces globally optimal solutions. However, this technique does not consider coupling constraints – for example, a problem wherein one agent's assignment directly affects the domain of feasible assignments for other agents, as is the case when agents are performing tasks subject to temporal and resource constraints.

Chien et al. proposed planning methods for a team of robotic rovers to accomplish a set of scientific objectives [48]. The rovers had to negotiate shared resources in order to accomplish their tasks. This approach uses an iterative-repair centralized planner coupled with an auction algorithm to perform centralized goal allocation and decentralized route planning and goal sequencing. Chien et al. benchmarked against a set of randomized problems with three rovers and 12 goals; however, thorough empirical evaluation with an optimal benchmark is not reported.

Lemaire et al. approached the problem of allocating multiple UAVs to perform a set of tasks where the task set was represented as a bipartite graph [151]. Here, one set of tasks (UAV navigation) was required to be completed before the second set (target sensing). The authors first presented a centralized auction solution and showed empirical results for a problem involving 50 tasks and four agents. Next, they described a distributed approach wherein an auctioneer agent assigns the first set of tasks to the multi-robot team, and then the second set of tasks is auctioned. This method supports rescheduling in light of dynamic disturbances occurring during task execution; however, the authors did not report empirical results for their distributed method.

As reported by Korsah, Stenz, and Dias, there is no well-known, general mathematical definition in the combinatorial optimization literature for problems within the XD [MT-MR-TA] category class. However, there is some work in application-specific solution techniques. Jones et al. consider a disaster response scenario where a set of fires must be extinguished by firefighters. Some routes to the fires are blocked by rubble, which may be cleared by bulldozers. Thus, one must decide which roads are to be cleared and, then, how to optimally plan a schedule for the firefighters to extinguish the fires [121]. To solve this problem, Jones et al. present both a multi-tiered auction approach and a genetic algorithm. This problem is slightly easier in that each agent can only perform one task, and each task only requires one agent.

Another auction-based approach, called TraderBots, is proposed by Dias [62]. In this work, agents can bid on *task trees*, or possible task decompositions, rather than only on simple, fully decomposed tasks. Agents may also bid on decompositions suggested by other agents. The agents, or bots, then negotiate, form centralized subgroups, and self-organize to select an assignment of agents to tasks and decompositions of those tasks to reduce the plan cost. For problems with tasks requiring multiple agents, Tang and Parker present an approach, ASyMTRe, for coalition formation where only the immediate assignment of agents to tasks is required (i.e., not a time-extended schedule). ASyMTRe is a centralized approach where each robot uses network *schemas* to solve the coordination problem [244]. These schemas consist of a list of input and output variables and an associated behavior that maps inputs to outputs. ASyMTRe then greedily connects the input and outputs across the robots to form coalitions which can act together to complete the task [244].

Finally, Nunes et al. developed the Temporal Sequential Single-Item auction (TeSSI) algorithm for decentralized task allocation and sequencing [188]. This technique takes as input a task set in the form of a simple temporal problem. Each task within the set has an earliest start time and latest finish time (absolute wait and deadline constraints). Constraints relating tasks are comprised exclusively of travel time constraints; resource constraints are not included. Nunes et al. [188] empirically demonstrated that their approach yields improved performance over CBBA [32]. More recently, their approach has been extended to handle precedence relations among tasks [169]: TeSSI and its variants [169, 188] solve a narrower class of problems than Tercio in that they do not handle cross-schedule dependencies, task-task deadlines, or resource capacity constraints. However, TeSSI and its variants distinguish themselves from Tercio [91] in that they solve this narrower class of problems in a decentralized fashion.

2.2.3 Hybrid Solution Techniques

Other hybrid approaches integrated heuristic schedulers within the MILP solver to achieve better scalability characteristics. For example, Chen et al. incorporated depth-first search (DFS) with heuristic scheduling [43]. In this approach, a DFS algorithm sequentially assigns tasks to agents (i.e., resources), and a heuristic scheduling algorithm sequences the tasks according to a minimum slack priority. The algorithm also employs heuristics to guide the order in which tasks are assigned to resources during the search. Chen et al. benchmarked their work on problems involving approximately 50 tasks and 10 resources (or agents) using a standard problem database [69, 201].

Alternatively, Castro et al. [40] used a heuristic scheduler to seed a feasible schedule for the MILP with regard to patient procedures conducted within a hospital. This method incorporates a tiered approach to minimize a three-term objective function: First, a heuristic scheduling algorithm generates an initial feasible solution. Next, the MILP is solved, with the first term of the objective function using the heuristic solution as a seed schedule. Then the MILP is solved again to optimize the second objective function term, using the solution from the first MILP as a constraint. The fourth step repeats this process, but for the third objective function term. The solution time is reduced by sequentially optimizing the objective function terms; however, this approach sacrifices global optimality.

Gombolay et al. developed a human-robot scheduling algorithm, Tercio, which leverages an analytical schedulability test to improve computational efficiency [91]. Tercio is an iterative algorithm. In each iteration, Tercio performs task allocation using a MILP. Given this task allocation, Tercio uses a combination of a heuristic sequencer with a schedulability test to ensure schedule feasibility. Rather than using search to find a satisficing schedule, Tercio uses the polynomial-time, analytical, schedulability test before each scheduling assignment is made to determine whether the assignment will result in infeasibility. Tercio is able to find solutions empirically within 10% of the optimal solution and scale to solve problems with tens of workers and hundreds of tasks in minutes [91].

Other approaches perform cooperative scheduling by incorporating Tabu Search within the MILP solver [242, 243], or by applying heuristics to abstract the problem to groupings of agents [141]. These hybrid methods are able to solve scheduling problems involving five agents (or groups of agents) and 50 tasks in minutes or even seconds, and address problems incorporating multiple resources, task precedence and temporal constraints relating task start and end times to the plan epoch time. However, these approaches do not take more general, task-task temporal constraints into consideration.

2.2.4 Conclusion

The approaches presented here commonly employ greedy techniques to solve tasks with complex dependencies. This tendency to rely on heuristic decomposition techniques increases as the problem becomes more computationally complex. Exact mathematical formulations can only scale to consider five workers and fifty tasks [107] and do not scale well to solve problems with complex dependencies [139]. To scale beyond these problem sizes within a more simple class of problems (i.e., ND and ID), or to coordinate workers performing tasks with complex interdependencies or conditional constraints (i.e., XD and CD), one must rely more heavily upon clever heuristic techniques.

Relying on hand crafted heuristic techniques for specific problems is not a scalable approach. The researcher must embed him or herself within each problem domain, attempt to solicit knowledge from the problem's practitioners, and develop a solution technique. However, these practitioners and domain experts are not readily able to provide this knowledge in the form of a codifiable strategy [45, 206]. To be able to develop a more general scheduling framework, we must be able to learn these strategies directly from the domain expert's demonstrations.

2.3 Learning from Human Demonstration

LfD is an active subfield of machine learning [2, 19, 115, 137, 269, 189, 247, 248, 252, 270]. Arguably, the most ubiquitous approach to LfD is Inverse Reinforcement Learning (IRL). IRL is founded on a Markov Decision Process $M = (S, A, T, \gamma, R)$ where:

- S is a set of states.
- A is a set of actions.

- $T : S \times A \times S \rightarrow [0, 1]$ is a transition function where $T(s, a, s')$ is the probability of begin in state s' after executing action a in state s.

- R: $S \rightarrow \mathbb{R}$ ($S \times A \rightarrow \mathbb{R}$) is a reward function that takes the form of $R(s)$ or $R(s, a)$ depending on whether the reward is assessed for being in a state or for taking a particular action in a state.

- $\gamma \in [0, 1)$ is the discount factor for future rewards.

In a Markov Decision Process, the goal is to learn a policy $\pi : S \rightarrow A$ that dictates which action to take in each state to maximize the infinite-horizon expected reward starting in state s. This reward is defined by a value function, $V^\pi(s)$, as shown in Equation 2.5. This equation states that the the expected reward one receives from following a trajectory given by π is a sum of all the rewards collected at each state along that trajectory, discounted by how many steps are required to reach those states.

$$V^\pi(s) = \mathbb{E}_\pi \left[\sum_{t=0}^{T} \gamma^t R(s_t) | s_o = s \right] \tag{2.5}$$

The value function satisfies the Bellman Equation for all $s \in S$, as shown in Equation 2.6. In essence, this equation sates that the total reward of being in state s is equal to the immediate reward available at s as well as a discounted future reward expected if one followed the path defined by π upon leaving state s.

$$V^\pi(s) = R(s) + \gamma \left[\sum_{s' \in S} T(s, \pi(s), s') V^\pi(s') \right] \tag{2.6}$$

A policy π is an optimal policy π^* *iff* $\forall s \in S$ Equation 2.7 holds. In other words, the optimal policy ensures that each action one takes is guaranteed to maximize the expected reward at the next state as well as all future states traversed.

$$\pi(s) = \operatorname*{argmax}_{a \in A} \left(\sum_{s' \in S} T(s, a, s') \left(R(s') + \gamma V^\pi(s') \right) \right) \tag{2.7}$$

The problem of IRL is to take as input 1) a Markov Decision Process (MDP) without a known Reward Function R, and 2) a set of m expert demonstrations $O = \{(s_o, a_o), (s_1, a_1), \ldots, (s_m, a_m)\}$ and then determine a reward function R that explains the expert demonstrations. The computational bottleneck of IRL and dynamic programming, in general, is the size of the state-space. The algorithms that solve these problems typically work by iteratively updating the estimate of the future expected reward of each state until convergence. For many problems of interest, the number of states is too numerous to hold in the memory of modern computers, and the time required for the expected future reward to converge can be impractical.

Researchers have extended the capability of IRL algorithms to be able to learn from operators with differing skill levels [207] and identify operator subgoals [172]. IRL is able to leverage the structure of the MDP to bind the rationality of the agent. Other researchers have investigated how robots can learn from demonstration via reinforcement learning, where the operator provides real-time feedback for the reward an agent should receive at each location in the state space [187, 248]. Nikolaidis and Shah use reinforcement learning and an approach, called *cross-training*, in which the human and robot switch roles on a team to improve the rate of learning [187]. Thomaz and Breazeal have shown how humans teach robots by providing feedback as a reward signal not just for the current state of the robot but also for anticipated future states; they extend a reinforcement learning framework to handle the anticipatory feedback from human instructors [248]. However, resource optimization or scheduling is highly non-Markovian: the next state of the environment is dependent upon the history of actions taken to arrive at the current state and the current time. Some researchers have tried to extend the traditional Markov Decision Process to characterize temporal phenomena, but these techniques do not scale up efficiently [27, 57, 263].

2.3.1 Recommender/Preference-Learning Systems

While not typically considered as LfD, the study of recommender systems is an important area of consideration in the vein of learning goals for the sake of recommending actions. Recommender systems have become ubiquitous in the Internet age with services such as Netflix [138]. One can think of recommender systems is learning how to recommend a series of movies, for example, to maximize the viewer's entertainment. These systems generally fall into one of two categories: collaborative filtering (CF) and content-based filtering (CB) [199]. In essence, collaborative filtering is a technique in which an algorithm learns to predict content to a single user based upon that user's history and that of other users who share common interests [199]. However, CF suffers from sparsity and scalability problems [199]. Content-based filtering works by comparing the similarity between content that the user has previously viewed and new content [53, 102, 223]. The challenge of content-based filtering lies in the difficulty in measuring the similarities between two items, and these systems can often over-fit, only predicting content that is very similar to what the user has previously used [15, 225]. Researchers have employed association rules [49], clustering [157, 158], decision trees [134], k-nearest neighbor algorithms [133], neural networks [7, 114], link analysis [34], regression [167], and general heuristic techniques [199] to recommend content to users.

Ranking the relevance of web pages is a key focus within systems that recommend suggested topics to users [36, 97, 100, 120, 193, 194, 156, 251, 253]. The seminal paper on page ranking, by Page et al., started the computational study of web page ranking with an algorithm, PageRank, which assesses the relevance of a web page by determining the number of pages that link to the page in question [193]. Since this paper, many have focused on developing better models for recommending web pages to users, which can then be trained using various machine learning algorithms [97, 100, 120, 194]. Within this discipline, there are three primary approaches to modeling the importance of a web page: point-wise, pair-wise, and list-wise ranking. In point-wise ranking, the goal is to determine a score, via regression analysis, for a web page given features describing its contents [156, 193]. Pair-wise ranking is typically a classification problem where the aim is to predict whether one page is more important than another [120, 194]. More recent efforts have focused on list-wise ranking, where researchers develop loss-functions based on entire lists of ranked web pages rather than individual pages or pair-wise comparisons between pages [36, 251, 253].

Researchers in scheduling have also tried to create an autonomous scheduling assistant that can learn the preferences of the user [18, 19]. Berry et al. produced a number of works spanning a decade [18, 19] which served to develop an automated scheduling assistant called PTIME. The purpose of PTIME was to help human co-workers schedule meetings. Berry et al. used extensive questionnaires to solicit the preferences of human workers for how they like to arrange their calendar. PTIME would take these preferences as input and map it to a mathematical objective function. When a new meeting needed to be arranged amongst the workers, PTIME would solve a mixed-integer mathematical program to determine the optimal time for the meeting to occur. However, after approximately a decade of work, the ultimate acceptance rate of PTIME's suggestions was only 60%. These authors conducted a retrospective analysis of their work and presented the following two tenets for future researchers to hold true.

1. "A personal assistant must build trust." [19]

2. "An assistive agent must aim to support, rather than replace, the user's natural process." [19]

These tenants have served as an inspiration for this thesis, and I believe all future works should start from these key design principles [18, 19].

Another example is work by De Grano et al., who present a method to optimize scheduling shifts for nurses by soliciting nurses' preferences through an auction process [93]. Other work in scheduling preferences has focused on developing techniques to efficiently generate schedules once the preferences have been solicited [232, 260].

One key area of focus is in modeling preferences as a set of *ceteris paribus* (all other things being equal) preference statements [25, 26, 192]. In this work, researchers solicit preferences from users typically in the form of binary comparisons. For example, consider the problem of determining which food and drink to serve a guest [25]. You may know the following:

- The guest prefers to drink red over white wine when eating a steak.

- The guest prefers steak over chicken.
- The guest prefers white wine when eating chicken.

Interestingly, determining the optimal food-drinking pairing can be found in polynomial-time; however, determining the relative optimality of one pairing over another (known as dominance testing) is NP-complete [25].

2.3.2 Challenges with Reward/Preference Learning

One limitation with reward learning is that it assumes an efficient mechanism to learn a policy to maximize the reward function. In the case of IRL, one relies on reinforcement learning (RL). However, exact RL requires enumerating and exploring an intractably large state space [262, 255, 268]. Considering just the decision variables for assigning a agents to n tasks, and ordering those tasks in time, the computational complexity is $O(2^{an}n!)$, which is computationally intractable for many scheduling problems. Yet, even if one approximately solves the RL problem [137, 236], RL is still ill-suited for handling the temporal and spatial task dependencies inherent in scheduling problems. The little work that has been done in scheduling via RL assumes models that are too restrictive: tasks must be periodic, occurring with a regular frequency, and independent, meaning there are no dependencies between tasks.

Methods specific to scheduling still suffer from issues with computational tractability. As mentioned previously, Berry et al. used a preference learning algorithm to codify an objective function, which could then be solved by mathematical optimization [18]. Similarly, Wilcox et al., use mathematical programming to maximize users' scheduling preferences [260]. As detailed in Section 2.2, exact mathematical optimization is not a scalable technique for complex scheduling problems. To solve these more complex problems, we need efficient heuristic methods to reduce the search space. What is needed is a mechanism to automatically learn a policy to intelligently explore the search space, reducing computation time.

2.3.3 Policy/Imitation Learning

A promising approach, called policy or imitation learning, focuses on directly learning a mapping from states to actions rather than learning a goal and using state-space enumeration and exploration to construct a suitable [46, 112, 220, 208]. This technique has been applied to learn cognitive decision-making tasks from human experts, such as determining an airport runway configuration [208]. Similarly, the learning system AlphaGo incorporates an initial policy-leaning phase [230]. The AlphaGo framework began by solving a supervised policy learning problem to imitate the decision-making of human Go players.

Chernova and Veloso developed a policy-learning approach in which a Gaussian Mixture Model is first used to learn a reasonable policy for a given task (e.g., driving a car on a highway). Then, the algorithm uses that knowledge to solicit user feedback by constructing scenarios where there is a high level of uncertainty. Chernova and Veloso also explore a second policy-learning approach using support vector machines to learn when an autonomous agent should request more demonstrations [47].

Sauppé and Mutlu study the problem of learning a model for the human behavior of back-channeling, which is a form of communication by an addressee to facilitate turn-taking and acknowledge speakership [247]. They demonstrate that a regression-based approach can predict the exhibition of these behaviors [247]. In more recent work, Huang and Mutlu study how humans employ multi-modal communication behaviors to present information in the form of a story [112]. They note that previous attempts at modeling typically employ a laborious process of hand-crafting rules and heuristics that lack generality [112]. Huang and Mutlu develop a robotic platform that uses a dynamic Bayesian network to learn how people choreograph their speech, gaze, and gestures for narration [112].

Ramanujam and Balakrishnan investigate the problem of learning a discrete-choice model for how air traffic controllers decide which set of runways to use for arriving and departing aircraft based on weather, arrival and departure demand, and other environmental factors [208]. Ramanujam and Balakrishnan train a discrete-choice model on real data from air traffic controllers and show how the model can accurately predict the correct runway configuration for the airport [208].

Sammut et al. apply a decision tree model for learning an autopilot, which autonomously controls an aircraft, from expert demonstration [220]. Their approach generates a separate decision tree for each of the following control inputs: elevators, ailerons, flaps, and thrust. In their investigation, they note that each pilot demonstrator could execute a planned flight path differently. These demonstrations could be in disagreement, thus making the learning problem significantly more difficult. To cope with the variance between pilot executions, a separate model was learned for each pilot. Inamura et al. use a Bayesian Network to learn which behavior to use and when to use that behavior via demonstration and interaction with that demonstrator [116]. Saunders et al. use a k-Nearest Neighbors approach with feature selection to learn which features to use in each scenario to predict the appropriate action [224]. In [218], Rybski et al. employ a Hidden Markov Model to learn which of a pre-learned set of actions must be applied and in what order to complete a demonstrated box-sorting task.

While these techniques have not typically been applied to scheduling, I believe policy learning is the right first step in learning from human demonstration how to schedule.

2.3.4 Conclusion

Policy learning is a promising approach for solving scheduling problems by learning policies through expert demonstration. Goal or reward learning approaches are able to capture high-level goals to produce quality schedules [2, 18]. However, these approaches are limited by their reliance on computational methods of exploring the search space to find a quality schedule. In the case of IRL, one relies on dynamic programming, which requires state-space enumeration. In the case of approaches such as PTIME [18], one relies on pure mathematical programming. One needs a method of intelligently isolating the important subspace where high-quality solutions exist to reduce the associated computation time.

Policy learning, on the other hand, is specifically designed to explore a state space. With a function mapping states to actions, one can theoretically construct a schedule through taking sequential scheduling actions (e.g., assigning a worker to a task at the present time). The challenge, however, is that I am unaware of any prior attempts to apply policy learning to the scheduling domain. Thus, a core aim of my thesis is to develop such a method: apprenticeship scheduling.

2.4 Blending Machine Learning and Optimization

Typically, reward and policy learning are limited by the quality of the demonstrations. Yet, even if the demonstrations are high quality, one cannot assume demonstrators nor their demonstrations will be optimal or even uniformly sub-optimal [5, 220]. As such, some have sought to directly model the sub-optimality of the demonstrations. For example, Zheng et al. make a clever extension to the work of Ramachandran and Amir to model the trustworthiness of the demonstrator within a softmax formulation transition function for reinforcement learning [269], as shown in Equation 2.8. In this equation, $Q^{\pi^*(R)}(s,a)$ is the expected reward of taking action a in state s assuming reward function $\boldsymbol{R}$ with the associated optimal policy π^*.

$$Pr((s,a)|\alpha;\boldsymbol{R}) = \frac{e^{\alpha Q^{\pi^*(\boldsymbol{R})}(s,a)}}{\sum_{a'} e^{\alpha Q^{\pi^*(\boldsymbol{R})}(s,a')}} \tag{2.8}$$

Through such a mechanism, it is possible to learn a policy that outperforms the human demonstrators by inferring the intended goal rather than the demonstrated goal. Zheng et al. showed that their approach was better able to capture the ground-truth objective function from imperfect training data than regular IRL [207], which does not include a trustworthiness parameter for demonstrations. Zheng et al. validated their approach using a synthetic data set in an experiment with the goal of identifying the best route through an urban domain. However, a limiting assumption is that one is able to accurately measure the trustworthiness of the demonstrations.

AlphaGo is a strong, recent advance in AI for its ability to play a turn-based strategy game, Go, at a super-human level [230]. AlphaGo is a multi-tier machine learning-optimization framework. Yet, at its core, AlphaGo is based on policy learning. Alpha Go uses Monte-Carlo Tree Search (MCTS) that is guided by a neural network policy trained on a data set of thirty million examples of human Go expert demonstrations [230].

While policy π chooses how to initially explore the search tree, AlphaGo employs two additional components to evaluate the quality of each branching point in the tree. The first component is a second policy, π', which is identical to the first except that the neural network is composed of fewer nodes. This smaller size enables the second policy to rapidly play the Go game to completion to predict a winner [230].

The second component is a value function trained via q-learning. The developers of AlphaGo rewired and duplicated the initial policy π to enable improvement through self-play. These duplicated, rewired policies $\pi_{Self-Play}$ would play Go against each other ad nauseum and, at each iteration, employ policy gradient to improve their policies [236]. The AlphaGo developers then captured a data set of thirty million moves taken by these policies. This data set was then used to train a q-learning algorithm to predict the expected value of taking a given action in a given state. Importantly, the AlphaGo authors note that these self-play policies actually performed worse than the original π trained on actual human demonstrations. Yet, they have not developed a cohesive theory for why π performs better [236]. Nonetheless, AlphaGo serves as a key example for how policy learning, coupled with optimization techniques (e.g., q-learning and policy gradient) can yield super-human solution quality.

Relevant to the problem of bridging machine learning and optimization, Banerjee et al. consider a scheduling problem for aircraft carrier flight deck operations. In this domain, the crew must repeatedly solve a scheduling problem wherein the variables remained the same (i.e., variables describing which workers perform which tasks and when), but the constraints (i.e., temporal constraints relating those tasks) for those variables changed [11]. Using a MILP formulation, they proposed a machine learning-optimization pipeline in which the system performed a branch-and-bound search over the integer variables, and used the prediction of a regression algorithm trained on examples of previously solved problems to provide a provable lowerbound on the optimality of the current integer variable assignments. A shortcoming of this approach is its reliance upon the ability to generate a large database of solutions to train the regression algorithm. This generation requires the costly exercise of repeatedly solving a large set of MILPs, which can be intractable for large-scale scheduling problems.

2.4.1 Conclusion

In Section 2.3, I presented the capabilities of reward/preference learning. Techniques, such as IRL, can take as input a human demonstration, learn a high-level goal based on those demonstrations, and generate a policy to maximize that goal. However, generating that policy is computationally challenging. Specifically, one needs clever heuristics to narrow the search space to find the goal.

On the other hand, learning the heuristics directly via policy learning can make it possible to compute high-quality solutions. Yet, the approaches discussed above (i.e., policy and imitation learning) suffer from the problem of dead reckoning. As a schedule is sequentially computed, even small deviations can add up to result in a large, cumulative deviation from the goal: an optimal schedule. Thus, we need the ability to both learn a policy and reason about the goal to generate optimal solutions.

There has been some initial work in the field of learning a policy from demonstration and using that policy to create solutions better than those demonstrations. In the case of Zheng et al., the authors attempt to infer the intended demonstration from an imperfect one [269]. In the case of AlphaGo, the authors use policy gradient and q-learning to construct a MCTS algorithm based on an initial, policy learning approach [230]. However, these approaches are not well-suited to scheduling for the reasons outlined in Section 2.3.

In the case of Banerjee et al., the authors use an approach more amenable to scheduling, mathematical programming, to provide optimal scheduling demonstrations. The authors then use machine learning to predict the optimality of a partial solution to reduce the computation time of solving future problems. However, their approach relies on being able to enumerate an initial training data set of optimal solutions.

In this thesis, I develop a technique specifically suited for scheduling that is able to take imperfect scheduling demonstrations and generate optimal solutions.

2.5 Human Factors of Human-Robot Interaction

As the complexity of human-operated machines has increased, so has the need for increased machine autonomy in order to aid human operators. As such, researchers in the fields of human factors have sought to improve the fluency of the human-machine system. Here, I review related works and identify key gaps in the literature that demonstrate the need for the experimental investigation into embodiment, situational awareness, workload, and workflow preferences with an autonomous, robotic scheduling assistant.

2.5.1 Human-In-The-Loop Decision Making

The field of human factors has pursued efforts complementary to algorithm development, which focus on developing interfaces between human supervisors and the agents they are tasking [23, 79, 165, 219, 254]. The human-robot interface has long been identified as a major bottleneck for the utilization of these robotic systems to their full potential [39]. As a result, significant research efforts have been aimed at easing the use of these systems in the field, including the careful design and validation of supervisory and control interfaces [13, 55, 92, 109, 122].

Many researchers have focused on the inclusion of a human in the decision-making loop to improve the quality of task plans and schedules for robots or semi-autonomous systems [52, 55, 72]. This is particularly important if the human operators have knowledge of factors not explicitly captured by the system model or if scheduling decisions have life or death consequences. In a study of aircraft carrier flight deck operations, veteran operators used heuristics to quickly generate an efficient plan and outperformed optimization algorithms [217]. Other works aimed to leverage the strengths of both humans and machines in scheduling by soliciting user input in the form of quantitative, qualitative, hard, or soft constraints over various scheduling options. Recommended schedules were then autonomously compiled and provided to users [8, 94, 99, 164, 267]. Researchers also develop models for how people weigh exploit-versus-explore actions when seeking reward in an uncertain environment [212]. Within the medical domain, Szolovits et al. describe work in developing algorithms that mimic the reasoning of human domain experts [238] rather than explicitly codifying rules [239]. These attempts to model reasoning are better able to isolate a key set of possible diagnoses, utilize pathophysiologic reasoning, and model the complexities of the illnesses of specific patients [238].

Supervisory systems have also been developed to assist human operators in the coordination of the activities of either four-robot or eight-robot teams [42]. Experiments demonstrated that operators were less able to detect key surveillance targets when controlling a larger number of robots. Similarly, other studies have investigated the perceived workload and performance of subjects operating multiple ground mobile-based robots [3]. Findings indicated that a number of robots greater than two greatly increased the perceived workload and decreased the performance of the human subjects.

2.5.2 Aiding Humans via Autonomy

There has been a flourish of recent work focused on the development of an improved human-machine interface [13, 55, 66, 68, 92, 109, 122]. In such work, researchers often view the human operator as a vital component of the decision making loop, particularly when this operator has knowledge of factors that are difficult to capture within a manually-encoded, autonomous framework [52, 55, 72]. Complementary to approaches that include the human in-the-loop, other work has focused on development of computational methods able to generate scheduling solutions using information collected a priori from human experts [8, 94, 99, 164, 267].

Researchers have proposed various mechanisms for distributed decision making in the form of agents that can independently reason about their activities [31, 91, 121, 139, 188, 200, 246]. For example, Tellex, Knepper, et al. developed a system enabling a team of robots to autonomously perform assembly manufacturing tasks, asking a human worker for help only when needed. This system enables robots to make requests in a way that allows a human to easily comply with these requests [246]. Although, their work does not consider the challenges with the humans' cost of context switching and state of situational awareness. Nikolaidis et al. [187] developed a robotic system able to learn a mental model for how people perform various assembly manufacturing tasks and adapt workflow to improve fluency for a human-robot dyad.

Teleoperation and blended autonomy are another key area of investigation. Here, human and machine agents work jointly toward accomplishing physical actions as opposed to cognitive tasks [67, 101, 174, 202]. For example, Dragan et al. developed and explored an intelligent, customizable interface for teleoperation. This interface mediates the consequences of a human not being in close physical proximity to the action performed in order to make teleoperation more seamless, and leverages the autonomous capabilities of the robot to assist in accomplishing a given task [66]. Pierce et al. developed a data-driven method for learning a mapping of the arm motions necessary to reach a specific physical state (target pose) from a human's mental model and translating those motions to corresponding robot motions in a physical environment [202]. The robot is then able to use this learned mapping to aid the operator in achieving the desired robot pose [202].

Herlant et al. investigated the challenges of controlling a robotic arm using a low-dimensional input device, such as a joystick [101]. They showed that mode switching accounts for a significant time burden for the user, and developed an automatic mode selection algorithm that reduces this burden [101]. Muelling et al. [174] developed an improved brain-computer interface to alleviate the challenges of latency, low-dimensional user commands and asymmetric control inputs, all of which are common to robotic teleoperation. Their system relies upon combining computer vision, user intent inference and arbitration between the human and robotic systems. In their work,Muelling et al. [174] validated their system via experiments where participants used input from two intra-cortical implants to control a robotic manipulator with seven degrees of freedom. The researchers found that their brain-computer interface enabled completion of tasks that were previously infeasible without arbitration [174].

There is also evidence that the manner in which people receive and interact with machine autonomy is influenced by a number of additional factors, including individual differences among operators and system embodiment [105, 123, 135, 148, 149, 213, 240]. For example, Takayama and Pantofaru investigated proxemics – how a person maintains physical or psychological distance between himself and others – in human-robot interaction and found differences based on participants' gender and prior experiences interacting with robots and animals [240]. Takayama and Pantofaru found that a person's prior experience interacting with robots and animals, as well as that person's gender, affects how they want to physically interact with a robot [240]. Ju et al. studied the effect of embodiment to capture attention and engender a desire to interact with the system [123]. Lee et al. investigated the role of embodiment and tactile interaction in HRI [123]. Lee et al., found that embodiment positively affects a person's view of the agent; however, if the experimenter imposed a restriction on tactile interaction, embodiment would cause a null or negative effect [148]. Drury et al. develop a human-UAV awareness decomposition that enables a designer of human-UAV interaction to understand the situational awareness needs of human UAV operators. Working with a cohort of UAV operators, Drury et al. were able to associate specific components of their framework (e.g., poor design features limiting situational awareness) with incidents in which UAV operator trainees had difficultly completing their missions [70]. Nonetheless, there has been little study into the ways in which human factors considerations, including situational awareness, workload assignment, and workflow preferences must influence the design of computational techniques for mixed initiative human-robot teaming.

2.5.3 Situational Awareness

Within the field of human factors [74, 75, 77, 125, 214] – and, more recently, of human-robot interaction [44, 70, 81, 234] – the study of situational awareness has been of utmost importance, particularly in the context of aviation [179, 180, 181, 182, 183]. In her seminal paper [75], Endsley defined a three-level model for situational awareness:

perception (Level 1 SA), comprehension (Level 2 SA), and projection (Level 3 SA). These levels require the operator of a complex system to perceive the state of the environment, understand the meaning of this state, and project the state into the future to understand how that state must change [75].

In subsequent work, Kaber and Endsley explored varying levels of automation to test situational awareness. They found higher automation resulted in improved performance if the implementation of that automation did not fail; however, if implementation did fail, automation resulted in much poorer performance by the human operator. Also, they wrote, collaboration on a task (as opposed to a human or robotic agent performing a task alone) can result in poorer performance and less situational awareness [77].

Chen et al. outline the drawbacks to certain elements of teleoperation. The irony from their point of view is that humans typically need to teleoperate autonomous systems when the environment becomes most difficult;ideally, however, teleoperative systems should be designed to reduce workload as the environment becomes more difficult to navigate, not the other way around. As it stands now, the operator must gain a high situational awareness via a user interface viewed from a remote location [44]. Riley et al. focused on the value of telepresence. In this work, they found that situational awareness as well as subject attention are related to mode of presence (i.e., telepresence or in person) [214].

Kaber and Endsley attempted to address two design variables affecting situational awareness that previously had not been studied in conjunction: the level of automation and adaptive automation [125]. In adaptive automation, the allocation of tasks to a human and a machine changes as a function of the state of the environment [125]. Kaber and Endsley found that participants had higher situational awareness at lower levels of automation, and lower situational awareness at higher levels of automation. When adaptive automation changed such that participants experienced different automation levels at varied time spans, participants did not perceive the periods of higher automation as involving a smaller task load, as they were also monitoring the automated task execution [125]. Many researchers have focused on modeling situational awareness, yet few researchers have developed interfaces specifically designed to augment situational awareness [81].

One major gap in prior robotics and human factors literature is the study of situational awareness wherein humans plan and execute a sequence of actions collaboratively within a human-robot team. Much work has focused on the human in a supervisory control role (e.g., [77, 246]) or as part of a dyad, for which the coordination of actions is relatively simple [187].

2.5.4 Mental and Physical Workload

Workload is a key issue identified in human-subject literature, which has indicated that human performance is highly dependent upon workload [198, 204, 233, 249, 257]. To help evaluate mental workload, researchers have proposed various subjective and psycho-physiological metrics [29, 96, 140, 234]. The most well-known metric is the NASA Task Load Index (TLX): a subjective, multivariate means of evaluating perceived mental workload [96]. A combination of results from prior work has led to a model for the relationship between workload and performance: Workload that is too heavy or too light can degrade both performance and situational awareness [204, 249]. One of the consequences of a high workload is increased reliance upon and compliance with automation [197].

Researchers have previously sought effective means to reduce workload through the use of semi-autonomous decision support tools [227], particularly in the field of air traffic control, due to the notoriously challenging nature of aircraft coordination [161, 162, 185]. In work by Lokhande et al., air traffic controllers spent 81.9% of their time in a head-down position looking at information displays rather than performing the vital task of visually monitoring traffic on the ground, even with the aid of a decision support tool [162].

Loft et al. developed a model for predicting the level of workload for air traffic controllers, and confirmed results from prior work indicating that mental workload increases with task difficulty. However, they also observed an unexpectedly stronger effect on mental workload as a function of the ability of air traffic controllers to prioritize tasks and manage resources; controller's with superior abilities experienced a decrease in their mental workload [161].

While the relationship between workload and task performance has been studied extensively with regard to human factors, it remains uncharacterized in the context of human-robot teams in which a robotic agent plays a substantial role in coordinating physical work. Prior studies have shown that people prefer to delegate decision making about scheduling to a robotic agent [87], yet there is a gap in the literature regarding the effects of varying physical workload on team fluency in such a scenario.

2.5.5 Workflow/Scheduling Preferences

Researchers in the fields of AI and robotics have explored computational methods for incorporating preference-based constraints when coordinating human-robot teams [4, 18, 98, 142, 187, 259]. Wilcox et al. developed an adaptive preferences algorithm to dynamically schedule human-robot teams in real time according to the unique preferences of human workers, as human teams in a factory setting can vary greatly on how they accomplish assembly tasks [259]. Alami et al. encoded task-specific constraints and workflow preferences that allow for prediction of likely human actions [4]. Berry et al. developed an AI assistant, known as PTIME, to learn the preferences and schedule the activities of human operators via a mathematical programming technique [18]. Bayesian networks [142], first-order Markov logic networks, and AND-OR graphs [98] have also been used to predict human actions during human-robot collaboration.

Preferences for task scheduling have been the subject of prior study [93, 99, 163, 232], but the human factors of scheduling activities have not been as well assessed. Generally, research has focused on the implementation of fairness metrics (such as in the work of Zhang et al. [265]) and other mathematical formulations for optimally scheduling according to human team members' preferences [93, 99, 163, 232]. However, roboticists must also ask the fundamental question of whether these preferences should be included in robot decision making and, if so, how best to do so.

2.5.6 Embodiment

While the effects of embodiment on engagement in social judgment tasks are extensively studied and well-documented [130, 131, 240, 245], the relationship between embodiment and humans levels of trust and dependence is a relatively new area of research [10, 131, 154]. This topic is crucial if robots are to become more than companions, but advisors to people.

Trust is defined as "the attitude that an agent will help achieve an individual's goals in a situation characterized by uncertainty and vulnerability [145]," and dependence is a behavioral measure indicating the extent to which users accept the recommendation of robots or virtual agents. Measures of dependence are distinguished according to whether the user makes Type I or Type II errors [65]. "Type I" refers to reliance, or the degree to which users accept advice from an artificial agent when it offers low-quality recommendations. "Type II" refers to the extent to which human users reject advice from an artificial agent when the advice is of high quality. The degrees to which a user accepts high-quality advice and rejects low-quality advice are called "appropriate compliance" and "appropriate reliance," respectively.

Studies examining the effects of embodiment on trust and dependence necessarily include objective assessments of dependence and task performance in addition to subjective assessment of the user's trust in the system [10, 58, 131, 154, 195]. Scassellati et al. [10, 154] conducted a series of experiments to compare compliance rates when interacting with a physically embodied robot, a video of a robot, and a disembodied voice. The tasks involved users receiving instructions to move objects to different locations, along with strategy advice for solving Sudoku-like puzzles. The authors found that embodiment was associated with a higher rate of compliance with advice provided by the robot, and suggested this indicated a greater level of human trust for an embodied robot. Similarly, Kiesler et al. [131] found that participants consumed fewer calories after receiving health advice from a physically embodied robot, as compared with advice from a video of a robot or an on-screen animated virtual agent.

Studies in human factors and decision support indicate that increased anthropomorphism also affects user interactions. Pak et al. [195] evaluated how the anthropomorphic characteristics of decision support aids assisting subjects answering questions about diabetes influenced subjective trust and task performance. The results indicated that younger adults trusted the anthropomorphic decision aid more, whereas older adults were insensitive to the effects of anthropomorphism. Moreover, shorter question response time (after controlling for accuracy) was observed in both age groups, suggesting a performance gain when receiving advice from a more anthropomorphic aid. In another study, de Visser [58] varied the degree of anthropomorphism of a decision support system while participants performed a pattern recognition task. The results indicated that the perceived knowledgeableness of the system increased with increasing anthropomorphism; however, their findings on dependence were inconclusive.

The results from studies with embodied robots must be interpreted with caution since they were primarily focused on situations in which robots produced reliable and high-quality recommendations. There is a growing body of research indicating that the quality of decision support cannot be relied upon, especially during complex tasks [256]. Negative consequences of humans blindly depending upon imperfect embodied artificial intelligence have been previously reported [215]. For example, Robinette et al., conducted experiments in which a robot guided human participants during a mock emergency rescue scenario involving a building fire. All participants followed the robot, even when the robot led them down unsafe routes and/or displayed simulated malfunctions and other suspicious behavior [215]. In work by Bainbridge et al., users were more likely to pick up and throw away books, which one would not normally think to throw away, at a physical robots request rather than a request given by an virtual avatar [9].

Such dependence upon imperfect automation presents serious problems for robotic assistance during safety-critical tasks. This concern is heightened by results from studies indicating increased trust in and reliance upon embodied systems as compared with virtual or computer-based decision support, suggesting a higher possibility of committing Type I errors. However, we also note that prior studies on embodiment, trust, and dependence were conducted with novices rather than domain experts performing complex real-world tasks. This leaves us with founded concerns, but gaps in our understanding of how human-robot interaction impacts the decision making of expert resource nurses.

2.5.7 Conclusion

There has been a wealth of work in human factors showing the profound effect introducing automation into the human workspace can have on the human workers in that space. Research has shown that shifting decision-making authority from humans to robotic agents decreases situational awareness. Further, inappropriate levels of workload can decrease human task performance. However, these phenomena have not been studied in the context of mixed-initiative scheduling for human-robot teaming. Further, workflow and scheduling preferences have been identified as an important area of research. Yet, the effect of including or ignoring those scheduling preferences has similarly not been studied in the context of mixed-initiative scheduling for human-robot teaming.

Algorithmic embodiment and anthropomorphism has also been studied by human factors researchers. The common understanding is that embodiment and increasing anthropomorphism results in human team members over-trusting and over-relying on these robotic systems. However, these studies have primarily focused on situations in which the robot is assumed to produce high-quality solutions, and the human counterparts have typically been amateurs. The results of the limited work done thus far with experts shows that experts may not over-rely on an embodied or anthropomorphic system. Thus, we need to understand what the consequences of algorithmic embodiment are if we are to realize the vision of intelligent service robots in professional domains.

Chapter 3

Apprenticeship Scheduling: Learning to Schedule from Human Experts[1]

3.1 Introduction

For service robots to be able to integrate into the human workplace, these robots must be able to understand when, where, and how they should help, just as human apprentices are able to learn from masters the ability to adapt to their dynamic environment. These robots must have an understanding of how to choreograph the team's activities. However, scheduling is a challenging problem that affects almost every aspect of our lives. The problem of optimal task allocation and sequencing with upper- and lowerbound temporal constraints (i.e., deadlines and wait constraints) is NP-Hard (Bertsimas and Weismantel 2005), and real-world scheduling problems quickly become computationally intractable.

[1] This chapter is based upon work published at the International Joint Conference on Artificial Intelligence as well as Robotics: Science and Systems. The citations are as follows:
Gombolay, M., Jensen, R., Stigile, J., & Shah, J. (2016, July). Apprenticeship scheduling: Learning to schedule from human experts. In Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), New York City, NY, USA.
Matthew C. Gombolay, Xi Jessie Yang, Brad Hayes, Nicole Seo, Zixi Liu, Samir Wadhwania, Tania Yu, Neel Shah, Toni Golen, and Julie A. Shah (2016, June). Robotic Assistance in Coordination of Patient Care. In Proc. Robotics: Science and Systems (RSS), Ann Arbor, Michigan, USA.

Yet, there is hope. Human domain experts are able to learn from experience to develop strategies, heuristics and rules-of-thumb to effectively respond to these problems. The challenge I pose is to autonomously learn the strategies employed by these domain experts; this knowledge can be applied and disseminated more efficiently with this type of model than with a "single-expert, single-apprentice" model.

In this chapter, I propose a technique, which I call "apprenticeship scheduling," to capture this domain knowledge in the form of a scheduling policy. My objective is to learn scheduling policies through expert demonstration and validate that schedules produced by these policies are of comparable quality to those generated by human or synthetic experts. My approach efficiently utilizes domain-expert demonstrations without the need to train within an environment emulator. Rather than explicitly modeling a reward function and relying upon dynamic programming or constraint solvers, which become computationally intractable for large-scale problems of interest, my objective is to use action-driven learning to extract the strategies of domain experts in order to efficiently schedule tasks.

The key to my approach is the use of pairwise comparisons between the action taken (e.g., schedule agent a to complete task τ_i at time t) and the set of actions not taken (e.g., unscheduled tasks at time t), at each moment in time, to learn the relevant model parameters and scheduling policies demonstrated by the training examples. I validate my approach using a synthetic data set of solutions for a variety of scheduling problems, a real-world data sets of demonstrations from human military experts solving a variant of the weapon-to-target assignment problem [150], and a real-world data set of human nursing professionals coordinating the care of obstetrics patients in a labor and delivery unit in a hospital. The synthetic and real-world problem domains I use to empirically validate my approach represent two of the most challenging classes within the multi-agent task allocation taxonomy established by Korsah et al. [139].

3.2 Problem Domains

I aimed to empirically demonstrate the generalizability of my learning approach through application to a variety of problem types. Korsah et al. provided a comprehensive taxonomy for classes of scheduling problems, which vary according to formulation of constraints, variables. and objective or utility function [139]. Within this taxonomy, there are four classes addressing interrelated utilities and constraints: No Dependencies (ND) [160], In-Schedule Dependencies (ID) [31, 90, 188], Cross-Schedule Dependencies (XD) [91], and Complex Dependencies (CD) [121].

The Korsah et al. taxonomy also delineates between tasks requiring one agent, i.e., "single-agent tasks" (ST); and tasks requiring multiple agents, i.e. "multi-agent tasks" (MT). Similarly, agents that perform one task at a time are "single-task agents" (SA), and agents capable of performing multiple tasks at the same time are "multi-task agents" (MA). Lastly, the taxonomy distinguishes between "instantaneous assignment" (IA), in which all task and schedule commitments are made at the same time, and "time-extended assignment" (TA), in which current and future commitments are planned.

In this work, I demonstrate my approach for two of the most difficult classes of scheduling problems defined within this taxonomy: XD [ST-SA-TA] and CD [MT-MA-TA]. The first problem I consider is the vehicle routing problem with time windows, temporal dependencies and resource constraints (VRPTW-TDR), which is an XD [ST-SA-TA]-class problem. Depending upon parameter selection, this family of problems encompasses the traveling salesman, job-shop scheduling, multi-vehicle routing and multi-robot task allocation problems, among others. I next consider two problems within the more difficult CD [MT-MA-TA] class. The first is a complex variant of the weapon-to-target assignment problem (WTA) [150], known as ASMD. The second is that of operating as a *resource nurse*, the one nurse who is responsible for ensuring that the right patient is in the right type of room at the right time and that the right types of nurses are there to care for those patients.

3.2.1 Synthetic Data Set

In this section, I define the VRPTW-TDR problem used in the synthetic evaluation of my apprenticeship scheduling technique. The multi-agent coordination problem (with temporospatial constraints) can be readily formulated as a mixed-integer linear program (MILP). A solution to the VRPTW-TDR problem then consists of the assignment of tasks to agents and a schedule for each agent's tasks such that all constraints are satisfied and the objective function is near-minimized. Determining the optimal schedule subject to hard upper- and lowerbound temporal constraints is NP-hard [22].

In this formulation, $\boldsymbol{\tau}$ is the set of all tasks to be performed, and $\boldsymbol{A}$ is the set of agents a given to complete those tasks. $\boldsymbol{TC}$ is the set of Interval Temporal Constraints [59] relating tasks start and finish times. Upperbound temporal constraints are referred to as "deadlines", and lowerbound temporal constraints are referred to as "wait constraints." "Relative" wait constraints are denoted as $W^{rel}_{\langle\tau_i,\tau_j\rangle}$ and specify that τ_j starts at least $W^{rel}_{\langle\tau_i,\tau_j\rangle} \geq 0$ time units after τ_i ends. "Absolute" wait constraints $W^{abs}_{\tau_i}$ requires the start time of τ_i to occur at least $W^{abs}_{\tau_i}$ time units of the epoch start time. Relative deadline constraints are denoted as $D^{rel}_{\langle\tau_i,\tau_j\rangle}$ and specify that τ_j must finish within $D^{rel} \geq 0$ time units of the start of τ_i. Absolute deadline constraints $D^{abs}_{\tau_i}$ limit the finish time of τ_j to within $D^{abs} \geq 0$ time units of the epoch start time. $\boldsymbol{\tau_R}$ is the set of all task pairs $\langle\tau_i, \tau_j\rangle$ that are separated by less than the allowable spatial proximity. $l_i \in \mathbb{R}^2$ is the location of τ_i, and *speed* is the rate at which an agent can traverse $\mathbb{R}^2$. Finally, M is an artificial variable set to a large positive number and is used to encode conditional constraints.

Equation 3.1 minimizes the makespan (i.e., the time between the start of the first task and finish of the final task). Equation 3.2 ensures that each task is assigned to a single agent. Equation 3.3 encodes the explicit ordering of tasks according to the lowerbound, relative temporal constraints $W^{rel}_{\langle\tau_i,\tau_j\rangle}$. Equations 3.4 and 3.6 encode the minimum wait times and upperbound deadline constraints, respectively, between pairs of tasks. Likewise, Equations 3.5 and 3.7 encode the minimum wait times and upperbound deadline constraints, respectively, relative to the start of the schedule.

Equations 3.9 and 3.10 ensure that each agent only performs one task at a time as well as encoding the time required to travel between tasks. Equations 3.8 encode the agent-specific duration of each task, C_i^a. Equations 3.11 and 3.12 sequence actions to ensure that agents maintain safe buffer distances from one another while performing tasks. Note that Equations 3.11 and 3.12 couple the variables relevant to sequencing and spatial proximity constraints and to task start and end times, tight dependencies amongst agents' schedules.

$$\text{Objective:} \min z = \max_{\tau_i, \tau_j} (f_i - s_j) \tag{3.1}$$

subject to

$$\sum_{a \in \boldsymbol{A}} A^a_{\tau_i} = 1, \forall \tau_i \in \boldsymbol{\tau} \tag{3.2}$$

$$x_{\langle \tau_i, \tau_j \rangle} = 1, \forall W^{rel}_{\langle \tau_i, \tau_j \rangle} \in \boldsymbol{TC} \tag{3.3}$$

$$W^{rel}_{\langle \tau_i, \tau_j \rangle} \leq s_i - f_j, \forall W^{rel}_{\langle \tau_i, \tau_j \rangle} \in \boldsymbol{TC} \tag{3.4}$$

$$W^{abs}_{\tau_i} \leq s_i, \forall W^{abs}_{\tau_i} \in \boldsymbol{TC} \tag{3.5}$$

$$D^{rel}_{\langle \tau_i, \tau_j \rangle} \geq f_j - s_i, \forall D^{rel}_{\langle \tau_i, \tau_j \rangle} \in \boldsymbol{TC} \tag{3.6}$$

$$D^{abs}_{\tau_i} \geq f_j, \forall D^{abs}_{\tau_i} \in \boldsymbol{TC} \tag{3.7}$$

$$f_i - s_i \geq C^a_i - M\left(1 - A^a_{\tau_i}\right), \forall \tau_i \in \boldsymbol{\tau}, a \in \boldsymbol{A} \tag{3.8}$$

$$s_j - f_i \geq M\left(x_{\langle \tau_i, \tau_j \rangle} - 1\right) + M\left(2 - A^a_{\tau_i} - A^a_{\tau_j}\right) + \frac{\|l_i - l_j\|}{speed}, \forall \tau_i, \tau_j \in \boldsymbol{\tau}, a \in \boldsymbol{A} \tag{3.9}$$

$$s_i - f_j \geq -M x_{\langle \tau_i, \tau_j \rangle} + M\left(2 - A^a_{\tau_i} - A^a_{\tau_j}\right) + \frac{\|l_j - l_i\|}{speed}, \forall \tau_i, \tau_j \in \boldsymbol{\tau}, a \in \boldsymbol{A} \tag{3.10}$$

$$s_j - f_i \geq M\left(x_{\langle \tau_i, \tau_j \rangle} - 1\right), \forall \langle \tau_i, \tau_j \rangle \in \boldsymbol{\tau_R} \tag{3.11}$$

$$s_i - f_j \geq -M x_{\langle \tau_i, \tau_j \rangle}, \forall \langle \tau_i, \tau_j \rangle \in \boldsymbol{\tau_R} \tag{3.12}$$

The worst-case time complexity of a complete solution technique for this problem is dominated by the binary decision variables for task allocation ($A^a_{\tau_i}$) and sequencing ($x_{\langle \tau_i, \tau_j \rangle}$) and is given by $O\left(2^{an^3}\right)$, where a is the number of agents and n is the number of tasks. Agent allocation contributes $O\left(2^{an}\right)$, and sequencing contributes $O\left(2^{n^2}\right)$. In the manufacturing settings of interest, the number of tasks and tasks is typically much larger than the number of agents, so the computational bottleneck when solving for a schedule occurs within the sequencing sub-problem.

3.2.2 Anti-Ship Missile Defense

In ASMD, the goal is to protect one's naval vessel against attacks by anti-ship missiles using "soft-kill weapons" (i.e., decoys) that mimic the qualities of a target in order to direct the missile away from its intended destination.

Developing tactics for soft-kill weapon coordination is highly difficult due to the relationship between missile behavior and the characteristics of soft-kill weapons. The control laws governing anti-ship missiles are varied, and the captain must select the correct decoy types in order to counteract the associated anti-ship missiles. For example, a ship's captain may deploy a decoy that emits a large amount of heat to make an enemy heat-seeking missile fly toward the decoy rather than the ship. Also, an enemy missile may consider the spatial layout of all targets to select the nearest or furthest targets; the magnitude of the radar reflectivity, radar emissions, and heat emissions; or combinations there within.

Further, decoys have different financial costs and timing characteristics. Some decoys, such as unmanned aerial vehicles (UAVs), are able to function during the entire engagement, while others, such as infrared (IR) flares, evaporate after a certain time. In turn, a captain may be required to use multiple decoys in tandem in order to divert a single anti-ship missile. At the same time, a captain might be able to use a single decoy to defeat multiple missiles.

Moreover, there is a complex interplay between the types and locations of decoys relative to the control laws governing anti-ship missiles. For example, deployment of a particular decoy, while effective against one airborne enemy missile, may actually cause a second enemy missile that was previously homing in on a second decoy to now impact the ship when it would have missed otherwise.

The ASMD problem is characterized as the most complex class of scheduling problem according to the Korsah et al. taxonomy [139]: CD [MA-MT-TA]. The problem considers multi-task agents (MA) in the form of decoys, each of which can work to divert multiple missiles at the same time. The problem also considers multi-agent tasks (MT): a feasible solution may require the simultaneous use of multiple agents in order to complete an individual task. Further, time-extended agent assignment (TA) must be taken into consideration, given the potential future consequences of scheduling actions taken at the current moment. Finally, the ASMD problem falls within the CD class, because each task may be decomposed in a variety of ways – each with their own cost – to accomplish the same goal, and each decomposition affects the value and feasibility of the decompositions of other tasks.

ASMD Problem Formulation

In anti-ship missile defense, one must determine how to deploy a set of soft-kill weapons, or decoys, to prevent enemy anti-ship missiles from impacting one's own ship (Figure 3-1). These decoys represent the agents, and the neutralization of each missile represents a task. The effectiveness E_i^a of deploying a decoy a against target τ_i at a given location $\vec{x_a} = [x, y, \theta]$ and time t is dependent upon the time history of all other decoy deployments h. Decoys are able to distract many missiles (MT), and many decoys can be used to distract the same missile at various points of its trajectory. Task allocation and scheduling commitments are made over time (TA). The key challenge of this problem is that the time history of how decoys have been deployed thus far affects the future effectiveness of decoys, as well as where and when they should be deployed. Agents and tasks have defined starting locations. Each task (missile) is modeled as a dynamical system with a homing function $F_\tau(h, t)$ that guides the missile toward its target and is a function of the current time and the time history of previous decoy deployments. Decoys travel at a constant speed to their target locations x_g from the ship that deploys them.

I formulate the ASMD as a mixed-integer linear program in Equations 3.13 through

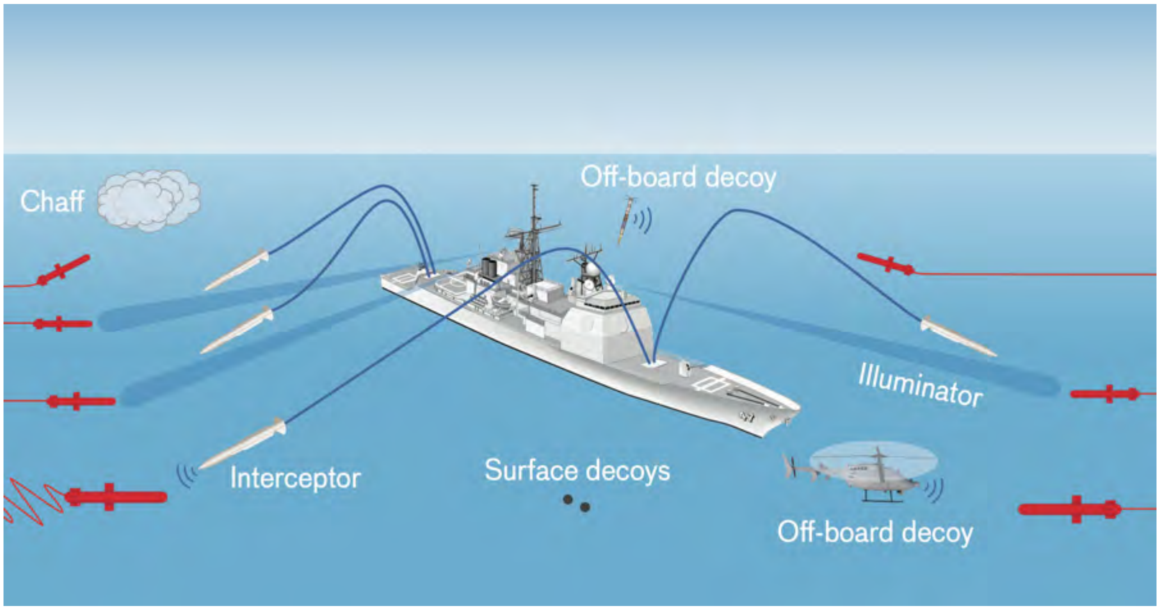

Figure 3-1: The tactical action officer aboard a naval vessel must coordinate a heterogeneous set of soft- and hard-kill weapons to defeat various enemy, anti-ship missiles.

3.34. This formulation incorporates a set of binary decision variables: $A_{d,m,t} \in \{0, 1\}$ is set to 1 to indicate that decoy d is assigned to missile m at time t, and is 0 otherwise. $A_{d,t} \in \{0, 1\}$ is set to 1 to indicate that decoy d is assigned to some missile at time t, and is 0 otherwise. $U_{d,m} \in \{0, 1\}$ is set to 1 to indicate that decoy d is used against missile m, and is 0 otherwise. $U_d \in \{0, 1\}$ is set to 1 to indicate that decoy d is used in the solution, and is 0 otherwise. $X_{d,l} \in \{0, 1\}$ is set to 1 to indicate that decoy d is deployed at location l, and is 0 otherwise. $V_m \in \{0, 1\}$ is set to 1 to indicate that missile m has been effectively diverted, and is 0 otherwise. $G_{g,m,t} \in \{0, 1\}$ is set to 1 to indicate that missile m is tracking the ship at time t. A single missile might have multiple, separate epochs during which it tracks the ship (e.g., it first tracks the ship, then tracks a decoy, then tracks the ship again after that decoy evaporates); thus, the program can choose which index g to represent the various epochs in $G_{g,m,t}$. $J_{d,m} \in \{0, 1\}$ is set to 1 to indicate that decoy d is deployed after missile m's flight (i.e., after it either hits the ship or is guided astray by a decoy).

The program contains the following continuous variables: $S^{decoy}_{d,m}$ represents the start time of the assignment of decoy d to missile m, and S^{decoy}_{d} is the time at which decoy d is deployed from the ship. Likewise, $F^{decoy}_{d,m}$ represents the finish time of the assignment of decoy d to missile m, and F_d^{decoy} is either the time at which the decoy evaporates or the end of the engagement. $S_{g,m}^{ship}$ indicates the start time of missile m tracking the ship during epoch g, and $F_{g,m}^{ship}$ indicates the finish time of missile m tracking the ship during epoch g.

The program also includes the following set of constants: $dt_m^{re-target}$ is the duration for which a missile will track a single target (i.e., decoy or ship) before re-assessing which target is best to track. Thus, if the missile begins tracking the ship at time t, no decoy can break its lock during the interval $[t, t + dt_m^{re-target})$. ETA_m is the time at which missile m will reach the ship's immediate vicinity. t_m^{appear} is the time at which missile m is first close enough to track the ship. c_d represents the financial cost of deploying decoy d. α, α', and α'' are predefined weighting terms for the objective function. The computational complexity of this formulation is dominated by the integer variables, which yields $O(2^{dmt+dm+dt+dl+d+gmt+m})$.

Equation 3.13 is a multi-criteria objective function that minimizes a weighted, linear combination of the cost of all decoy deployments, less the total time during which missiles are tracking decoys and the number of missiles successfully guided away from the ship.

Equations 3.14 and 3.21 ensure internal consistency between the variables. Equation 3.22 ensures that a decoy, if deployed, is active for dt_d^{evap} units of time given its timing characteristics. Equation 3.23 ensures that a decoy is deployed to no more than one location. Equation 3.24 ensures that, if a decoy is deployed against a missile, its deployment location will be a more attractive target for that missile than the ship. Equation 3.25 requires that each missile tracks either a ship or decoy while within range. Equations 3.26 and 3.27 force a decoy, if deployed, to a location that would cause missile m to impact the ship, to either be deployed after the missile has already been diverted or reached the ship (Equation 3.26) or to be deployed and evaporate before the missile enters targeting range (Equation 3.27).

Equation 3.28 ensures that a missile must be tracking a decoy in the final seconds before it reaches the vicinity of the ship, or else the missile will impact the ship. The duration of this critical period is dependent upon missile dynamics and the target selection process.

$$\min z,\; z = \alpha \sum_{d} c_d U_d - \alpha' \sum_{d,m,t} A_{d,m,t} - \alpha'' \sum_{m} V_m \tag{3.13}$$

$$A_{d,m,t} \leq A_{d,t}, \forall d, m, t \tag{3.14}$$

$$A_{d,m,t} \leq U_{d,m}, \forall d, m, t \tag{3.15}$$

$$X_{d,l} \leq U_d, \forall d, l \tag{3.16}$$

$$S_d^{decoy} \leq S_{d,m}^{decoy}, \forall d, m \tag{3.17}$$

$$S_{d,m}^{decoy} \leq t + M(1 - A_{d,m,t}), \forall d, m, t \tag{3.18}$$

$$F_{d,m}^{decoy} \leq F_d^{decoy}, \forall d, m \tag{3.19}$$

$$tA_{d,m,t} \leq F_{d,m}^{decoy}, \forall d, m, t \tag{3.20}$$

$$M(U_{d,m} - 1) \leq S_{d,m}^{decoy} - F_{d,m}^{decoy} - 1 + \sum_{t} A_{d,m,t} \leq M(1 - U_{d,m}) \tag{3.21}$$

$$M(U_d - 1) \leq F_d^{decoy} - S_d^{decoy} - dt_d^{evap} \leq M(1 - U_d) \tag{3.22}$$

$$\sum_{l} X_{d,l} \leq 1, \forall d \tag{3.23}$$

$$U_{d,m} \leq \sum_{l|m \text{ seduced by decoy } d \text{ in location } l} X_{d,l}, \forall d, m \tag{3.24}$$

$$1 = \sum_{d} A_{d,m,t} + \sum_{g} G_{g,m,t}, \forall m, t \tag{3.25}$$

$$\begin{gathered} t_m^{appear} - F_d^{decoy} \geq M(X_{d,l} + V_m - J_{d,m} - 2), \\ \forall d, l, m \text{ s.t. decoy d in location l would cause missile } m \text{ to impact the ship.} \end{gathered} \tag{3.26}$$

$$\begin{gathered} S_d^{decoy} - ETA_m \geq M(X_{d,l} + V_m + J_{d,m} - 3), \forall d, l, m \text{ s.t. decoy d} \\ \text{in location l would cause missile } m \text{ to impact the ship.} \end{gathered} \tag{3.27}$$

$$V_m \leq \sum_{d} A_{d,m,t}, \forall m, t | t \text{ in critical region for missile } m. \tag{3.28}$$

$$\begin{gathered} 2 \geq A_{d,m,t} + X_{d,l} + X_{d',l'}, \forall d, d', l, l', m, t \text{ s.t. missile m is more} \\ \text{attracted to decoy d' at location l' than decoy d at location l at time t.} \end{gathered} \tag{3.29}$$

$$1 \geq A_{d,m,t} + A_{d',m,t}, \forall d, d', m, t \text{ s.t. } d \neq d' \\ \text{and t is in a critical region before impact.} \tag{3.30}$$

$$S^{ship}_{g,m} \leq t + M(1 - G_{g,m,t}), \forall g, m, t \tag{3.31}$$

$$t * G_{g,m,t} \leq F^{ship}_{g,m}, \forall g, m, t \tag{3.32}$$

$$M(U_{g,m} - 1) \leq S^{ship}_{g,m} - F^{ship}_{g,m} - 1 + \sum_t G_{g,m,t} \leq M(1 - U_{g,m}) \tag{3.33}$$

$$\begin{aligned} F^{ship}_{g,m} - S^{ship}_{g,m} \geq\ & M(G_{g,m,t} - 1) \\ & + \begin{cases} dt^{re-target}_m - 1 & \text{if } t < ETA_m - dt^{re-target}_m, \\ ETA_m - t - 1 & \text{otherwise.} \end{cases} \\ & + \begin{cases} -MG_{g,m,t-1} & \text{if } t > t^{appear}_m, \\ 0 & \text{otherwise.} \end{cases} \\ & \forall g, m, t | t^{appear}_m \leq t < ETA_m \end{aligned} \tag{3.34}$$

Equation 3.29 ensures that a missile will select the most attractive decoy according to that missile's selection logic. Equation 3.30 restricts decoy deployments such that the missile heading does not "sweep" across the ship in the final seconds of the missile's flight. If a missile does not have enough time to change its direction toward a newly deployed decoy, that missile will fly into the ship.

Equations 3.31 through 3.34 ensure that the duration of epoch g of missile m while tracking the ship lasts exactly as long as the retargeting time for the missile. Equations 3.31 and 3.32 are akin to Equations 3.18 through 3.20 and relate the start and finish times of ship-tracking epoch g to the decision variable $G_{g,m,t}$. Equation 3.33 is akin to Equation 3.21 and relates the start and finish times of ship-tracking epoch g to the decision variable $G_{g,m,t}$. Equation 3.34 ensures that the tracking time is $dt^{re-target}_m$ if the missile is airborne for at least $dt^{re-target}_m$ seconds. Otherwise, the tracking time is equal to the time before impacting the ship (i.e., $ETA^m - t - 1$). Finally, a term (i.e., $-MG_{g,m,t-1}$) disables the constraint for all t except for the exact moment when t begins tracking the ship.

As ASMD is a time-extended problem, the formulation must discretize time. However, note that the granularity with which the task of protecting the ship from a given missile is decomposed as a function of time is a modeling choice with ramifications for the quality and computation time of a solution. Consider a missile that will hit a ship if it tracks a missile in some time interval $[t, t')$ for a duration $dt = t - t'$. The captain might, at time t, deploy a decoy d, such as a hovering UAV, that is able to last the entire duration dt. However, it may be preferable to deploy one or more decoys d', each of which remains active for a portion of the specified time interval. Furthermore, in a situation wherein another missile m' is launched before m, it may be best to have a decoy deployed before t that can divert both m and m' during part or all of those missiles' flights.

Because I do not know a priori the best time to deploy a decoy that can be used for varying portions (i.e., subtasks) of the task of mitigating each missile, I must decompose the task into sufficiently small time steps. Discretizing time exponentially increases the search space, and thus the time to compute the solution; therefore, there is a balance between optimality (and feasibility) and computation time. In order to generate an exact solution, I chose the least-common multiple of the time constants, which is trivially 1, as the unit of time in the simulation.

3.2.3 Labor and Delivery

This section provides a formal representation of the resource nurse's decision making problem. Section 5.3 describes how I implemented the decision support based on this formulation.

A resource nurse must solve a problem of task allocation and schedule optimization with stochasticity in the number and types of patients and the duration of tasks. A task τ_i represents the set of steps (i.e., subtasks) required to care for patient i, and each τ_i^j is a given stage of labor for that patient. Stages of labor are related by stochastic lower-bound constraints $W^{rel}_{\langle \tau_i^j, \tau_x^y \rangle}$, requiring the stages to progress sequentially. There are stochastic time constraints, $D^{abs}_{\tau_i^j}$ and $D^{rel}_{\langle \tau_i^j, \tau_x^y \rangle}$, relating the stages of labor to account for the inability of resource nurses to perfectly control when a patient will move from one stage of labor to the next. Arrivals of τ_i (i.e., patients) are drawn from stochastic distributions. The model considers three types of patients: scheduled cesarean patients, scheduled induction patients, and unscheduled patients. The set of $W^{rel}_{\langle \tau_i^j, \tau_x^y \rangle}$, $D^{abs}_{\tau_i^j}$ and $D^{rel}_{\langle \tau_i, \tau_j \rangle}$ are dependent upon patient type.

Labor nurses are modeled as agents with a finite capacity to process tasks in parallel, where each subtask requires a variable amount of this capacity. For example, a labor nurse may generally take care of a maximum of two patients. If the nurse is caring for a patient who is "fully and pushing" (i.e., the cervix is fully dilated and the patient is actively trying to push out the baby) or in the operating room, the nurse may only care for that patient.

Rooms on the labor floor (e.g., a labor room, an operating room, etc.) are modeled as resources, which process subtasks in series. Agent and resource assignments to subtasks are pre-emptable, meaning that the agent and resource assigned to care for any patient during any step in the care process may be changed over the course of executing that subtask.

In this formulation, ${}^tA^a_{\tau_i^j} \in \{0, 1\}$ is a binary decision variable for assigning agent a to subtask τ_i^j for time epoch $[t, t+1)$. ${}^tG^a_{\tau_i^j}$ is an integer decision variable for assigning a certain portion of the effort of agent a to subtask τ_i^j for time epoch $[t, t+1)$. ${}^tR^r_{\tau_i^j} \in \{0, 1\}$ is a binary decision variable for whether subtask τ_i^j is assigned resource r for time epoch $[t, t+1)$. $H_{\tau_i} \in \{0, 1\}$ is a binary decision variable for whether task τ_i and its corresponding subtasks are to be completed. $U_{\tau_i^j}$ specifies the effort required from any agent to work on τ_i^j. $s_{\tau_i^j}, f_{\tau_i^j} \in [0, \infty)$ are the start and finish times of τ_i^j.

$$\min fn \left(\{{}^tA^a_{\tau_i^j}\}, \{{}^tG^a_{\tau_i^j}\}, \{{}^tR^r_{\tau_i^j}\}, \{H_{\tau_i}\}, \{s_{\tau_i^j}, f_{\tau_i^j}\} \right) \tag{3.35}$$

$$\sum_{a\in A} {}^{t}A^{a}_{\tau_i^j} \geq 1 - M\left(1 - H_{\tau_i}\right), \forall \tau_i^j \in \boldsymbol{\tau}, \forall t \tag{3.36}$$

$$M\left(2 - {}^{t}A^{a}_{\tau_i^j} - H_{\tau_i}\right) \geq -U_{\tau_i^j} + {}^{t}G^{a}_{\tau_i^j} \geq$$
$$M\left({}^{t}A^{a}_{\tau_i^j} + H_{\tau_i} - 2\right), \forall \tau_i^j \in \boldsymbol{\tau}, \forall t \tag{3.37}$$

$$\sum_{\tau_i^j \in \boldsymbol{\tau}} {}^{t}G^{a}_{\tau_i^j} \leq C_a, \forall a \in A, \forall t \tag{3.38}$$

$$\sum_{r\in R} {}^{t}R^{r}_{\tau_i^j} \geq 1 - M\left(1 - H_{\tau_i}\right), \forall \tau_i^j \in \boldsymbol{\tau}, \forall t \tag{3.39}$$

$$\sum_{\tau_i^j \in \boldsymbol{\tau}} {}^{t}R^{r}_{\tau_i^j} \leq 1, \forall r \in R, \forall t \tag{3.40}$$

$$ub_{\tau_i^j} \geq f_{\tau_i^j} - s_{\tau_i^j} \geq lb_{\tau_i^j}, \forall \tau_i^j \in \boldsymbol{\tau} \tag{3.41}$$

$$s_{\tau_x^y} - f_{\tau_i^j} \geq W^{rel}_{\langle \tau_i, \tau_j \rangle}, \forall \tau_i, \tau_j \in \boldsymbol{\tau} |, \forall W^{rel}_{\langle \tau_i, \tau_j \rangle} \in \boldsymbol{TC} \tag{3.42}$$

$$f_{\tau_x^y} - s_{\tau_i^j} \leq D^{rel}_{\langle \tau_i, \tau_j \rangle}, \forall \tau_i, \tau_j \in \boldsymbol{\tau} | \exists D^{rel}_{\langle \tau_i, \tau_j \rangle} \in \boldsymbol{TC} \tag{3.43}$$

$$f_{\tau_i^j} \leq D^{abs}_{\tau_i}, \forall \tau_i \in \boldsymbol{\tau} | \exists D^{abs}_{\tau_i} \in \boldsymbol{TC} \tag{3.44}$$

Equation 3.36 enforces that each subtask τ_i^j during each time epoch $[t, t+1)$ is assigned one agent. Equation 3.37 ensures that each subtask τ_i^j receives a sufficient portion of the effort of its assigned agent a during time epoch $[t, t+1)$. Equation 3.38 ensures that agent a is not oversubscribed. Equation 3.39 ensures that each subtask τ_i^j of each task τ_i that is to be completed (i.e., $H_{\tau_i} = 1$) is assigned one resource r. Equation 3.40 ensures that each resource r is assigned to only one subtask during each epoch $[t, t+1)$. Equation 3.41 requires the duration of subtask τ_i^j to be less than or equal to $ub_{\tau_i^j}$ and at least $lb_{\tau_i^j}$ units of time. Equation 3.42 requires that τ_x^y occurs at least $W^{rel}_{\langle \tau_i^j, \tau_x^y \rangle}$ units of time after τ_i^j. Equation 3.43 requires that the duration between the start of τ_i^j and the finish of τ_x^y is less than $D^{rel}_{\langle \tau_i^j, \tau_x^y \rangle}$. Equation 3.44 requires that τ_i^j finishes before $D^{abs}_{\tau_i^j}$ units of time have expired since the start of the schedule.

The functions of a resource nurse are to assign nurses to take care of labor patients and to assign patients to labor beds, recovery room beds, operating rooms, ante-partum ward beds or post-partum ward beds. The resource nurse has substantial flexibility when assigning beds, and their decisions will depend upon the type of patient and the current status of the unit in question. They must also assign scrub technicians to assist with surgeries in operating rooms, and call in additional nurses if required. The corresponding decision variables for staff assignments and room/ward assignments in the above formulation are ${}^tA^a_{\tau_i^j}$ and ${}^tR^r_{\tau_i^j}$, respectively.

The resource nurse may accelerate, delay or cancel scheduled inductions or cesarean sections in the event that the floor is too busy. Resource nurses may also request expedited active management of a patient in labor. The decision variables for the timing of transitions between the various steps in the care process are described by $s_{\tau_i^j}$ and $f_{\tau_i^j}$. The commitments to a patient (or that patient's procedures) are represented by H_{τ_i}.

The resource nurse may also reassign roles among nurses. For example, a resource nurse may pull a triage nurse or even care for patients herself if the floor is too busy. If a patient's condition is particularly acute (e.g., the patient has severe pre-eclampsia), the resource nurse may assign one-to-one nursing. The level of attentional resources a patient requires and the level a nurse has available correspond to variables $U_{\tau_i^j}$ and ${}^tG^a_{\tau_i^j}$, respectively. The resource nurse makes his or her decisions while considering current patient status $\Lambda_{\tau_i^j}$, which is manually transcribed on a whiteboard, shown in Figure 3-2.

The stochasticity of the problem arises from the uncertainty in the upper and lowerbound of the durations $(ub_{\tau_i^j}, lb_{\tau_i^j})$ of each of the steps in caring for a patient, the number and types of patients $\boldsymbol{\tau}$, and the temporal constraints $\boldsymbol{TC}$ relating the start and finish of each step. These variables are a function of the resource and staff allocation variables ${}^tR^a_{\tau_i^j}$, ${}^tA^a_{\tau_i^j}$, and patient task state $\Lambda_{\tau_i^j}$, which includes information on patient type (i.e., presenting with scheduled induction, scheduled cesarean section, or acute unplanned anomaly), gestational age, gravida, parity, membrane status, anesthesia status, cervix status, time of last exam, and any co-morbidities. Formally, $\left(\{ub_{\tau_i^j}, lb_{\tau_i^j} | \tau_i^j \in \boldsymbol{\tau}\}, \boldsymbol{\tau}, \boldsymbol{TC}\right) \sim P(\{{}^tR^a_{\tau_i^j}, {}^tA^a_{\tau_i^j}, \Lambda_{\tau_i^j}, \forall t \in [0, 1, \ldots, T]\})$.

The computational complexity of satisfying constraints in Equations 3.36-3.44 is given by $O\left(2^{|A||R|T^2} C_a^{|A|T}\right)$, where $|A|$ is the number of agents, with each agent possessing an integer processing capacity of C_a; there are n tasks τ_i, each with m_i

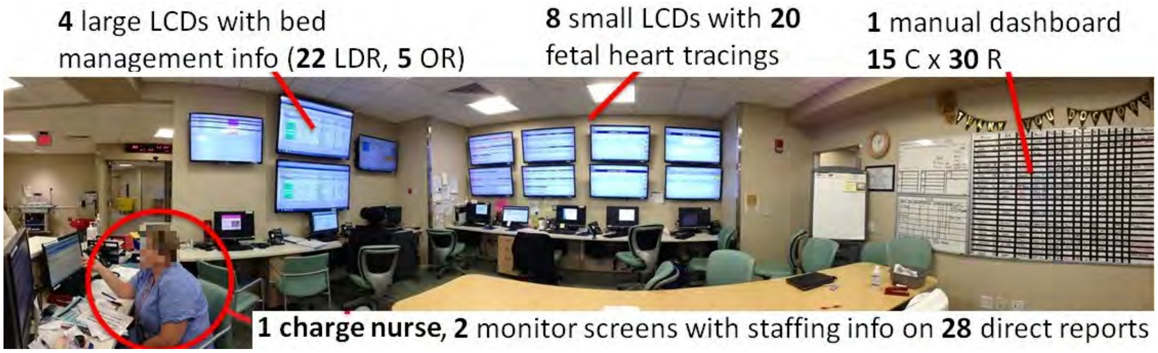

Figure 3-2: A resource nurse must assimilate a large variety and volume of information to effectively reason about resource management for patient care.

subtasks; $|R|$ resources; and an integer-valued planning horizon of T units of time. In practice, there are ~ 10 nurses (agents) who can care for up to two patients at a time (i.e., $C_a = 2, \forall a \in A$), 20 different rooms (resources) of varying types, 20 patients (tasks) at any one time and a planning horizon of 12 hours or 720 minutes, yielding a worst-case complexity of $\sim 2^{10*20*720^2} 2^{10*720} \geq 2^{10^6}$, which is computationally intractable for exact methods without the assistance of informative search heuristics.

3.3 Model for Apprenticeship Learning

In this section, I present a new computational method for learning, via expert demonstration, a scheduling policy that correctly determines which task to schedule as a function of the state of the scheduling environment. In Section 3.5, I apply this method to learn from demonstration how to make scheduling decisions in three practical domains.

Many approaches to learning via demonstration, such as reinforcement learning or IRL, are based on Markov models [14, 33, 136, 205]. Markov models, however, do not capture the temporal dependencies between states and are computationally intractable for large problem sizes. In order to determine which tasks to schedule at which times, I draw inspiration from the domain of web page ranking [193], or predicting the most relevant web page in response to a search query. One important component of page ranking is capturing how pages relate to one another as a graph with nodes (representing web pages) and directed arcs (representing links between those pages) [193]. This connectivity is a suitable analogy for the complex temporal dependencies (precedence, wait, and deadline constraints) relating tasks within a scheduling problem.

Within the three general classes of page-ranking models (i.e., pointwise [251], pairwise [120, 194], and listwise [36, 251, 253]), the pairwise model has key advantages. First, classification algorithms (e.g., support vector machines) can be directly applied [36]. Second, a pairwise approach is non-parametric, in that the cardinality of the input vector is not dependent upon the number of tasks (or actions) that can be performed in any instance. Third, training examples of pairwise comparisons in the data can be readily solicited. From a given observation during which a task was scheduled, I only know which task was most important – not the relative importance between all tasks. Thus, I create training examples based on pairwise comparisons between scheduled and unscheduled tasks. A pairwise approach is more natural because one lacks the necessary context to determine the relative rank between two unscheduled tasks.

Consider a set of tasks, $\tau_i \in \tau$, in which each task has a set of real-valued features, γ_{τ_i}. Each scheduling-relevant feature $\gamma^j_{\tau_i}$ may represent, for example, the deadline, the earliest time the task is available, the duration of the task, which resource r is required by this task, etc. Next, consider a set of m observations, $O = \{O_1, O_2, \ldots, O_m\}$. Observation O_m consists of a feature vector $\{\gamma_{\tau_1}, \gamma_{\tau_2}, \ldots, \gamma_{\tau_n}\}$ describing the state of each task, the task scheduled by the expert demonstrator (including a null task, $\tau_\emptyset$, if no task was scheduled) and the time at which an action was taken. The goal is to learn a policy that correctly determines which task to schedule as a function of the task state.

I deconstruct the problem into two steps: 1) For each agent/resource pair, determine the candidate next task to schedule; and 2) For each task, determine whether to schedule the task from the current state. Both of these abilities are learned through either observing a demonstrative scheduling action (e.g., assigning a task to an agent) or by observing that no scheduling action is taken (i.e., a null action). Being able to execute a null action, or idling, is necessary for solving scheduling problems with upper- and lowerbound temporal constraints [22]. For example, consider a task that must be executed at time $t + 1$ in order to satisfy its deadline; further, it's wait constraint will not be satisfied until that same time, $t + 1$. Yet, there is a second task that is available at the current time, t, and its deadline can be trivially satisfied by executing the task at any point during the schedule. An opportunistic scheduler would naturally schedule the second task because it is currently available; however, this would result in the first task violating its deadline constraint. Thus, the optimal decision would have been to idle for one unit of time before executing the first task.

In order to learn to correctly assign the next task to schedule, I transform each observation O_m into a new set of observations by performing pairwise comparisons between the scheduled task τ_i and the set of unscheduled tasks (Equations 3.45 and 3.46). Equation 3.45 creates a positive example for each observation in which a task τ_i was scheduled. This example consists of the input feature vector, $\phi^m_{\langle \tau_i, \tau_x \rangle}$, and a positive label, $y^m_{\langle \tau_i, \tau_x \rangle} = 1$. Each element of the input feature vector $\phi^m_{\langle \tau_i, \tau_x \rangle}$ is computed as the difference between the corresponding values in the feature vectors γ_{τ_i} and γ_{τ_x}, describing scheduled task τ_i and unscheduled task τ_x. Equation 3.46 creates a set of negative examples with $y^m_{\langle \tau_x, \tau_i \rangle} = 0$. For the input vector, I take the difference of the feature values between unscheduled task τ_x and scheduled task τ_i.

$$\begin{gathered} {}^{rank}\theta^m_{\langle \tau_i, \tau_j \rangle} := \left[\xi_{\boldsymbol{\tau}}, \gamma_{\tau_i} - \gamma_{\tau_j} \right], y^m_{\langle \tau_i, \tau_j \rangle} = 1, \\ \forall \tau_j \in \boldsymbol{\tau} \backslash \tau_i, \forall O_m \in \boldsymbol{O} | \tau_i \text{ scheduled in } O_m \end{gathered} \tag{3.45}$$

$$\begin{gathered} {}^{rank}\theta^m_{\langle \tau_j, \tau_i \rangle} := \left[\xi_{\boldsymbol{\tau}}, \gamma_{\tau_j} - \gamma_{\tau_i} \right], y^m_{\langle \tau_j, \tau_i \rangle} = 0, \\ \forall \tau_j \in \boldsymbol{\tau} \backslash \tau_i, \forall O_m \in \boldsymbol{O} | \tau_i \text{ scheduled in } O_m \end{gathered} \tag{3.46}$$

$$\widehat{\tau_i^*} = \underset{\tau_i \in \boldsymbol{\tau}}{\operatorname{argmax}} \sum_{\tau_j \in \boldsymbol{\tau}} f_{priority}\left(\tau_i, \tau_j\right) \tag{3.47}$$

$$
{}^{act}\phi^{m}_{\tau_i} := [\xi_{\boldsymbol{\tau}}, \gamma_{\tau_i}],
$$

$$
y^{m}_{\tau_i} = \begin{cases} 1 : \tau_i \text{ scheduled in } O_m \ \wedge \tau_i \text{ scheduled in } O_{m+1} \\ 0 : \tau_{\emptyset} \text{ scheduled in } O_m \end{cases} \tag{3.48}
$$

This feature set is then augmented to capture additional contextual information important for scheduling, which may not be captured in examples consisting solely of differences between task features. For example, a scheduling policy may change based on progress toward task completion; i.e., the proportion of tasks completed so far. To provide this high-level information, I include ξ_{τ}, the set of contextual, high-level features describing the set of tasks for observation O_m, in (Equations 3.45-3.46).

My technique relies upon the ability of domain experts to articulate an appropriate set of features for the problem. I believe this to be a reasonable limitation. Results from prior work have indicated that domain experts are adept at describing the (high-level, contextual and task-specific) features used in their decision-making; however, it is more difficult for experts to describe how they reason about these features [45, 206]. In future work, I aim to extend my approach to include feature learning rather (e.g., a convolutional neural network analogy for scheduling) than relying upon experts to enumerate the important features they reason about in order to construct schedules.

Given these observations O_m and their associated features, I can train a classifier $f_{priority}(\tau_i, \tau_j) \in \{0, 1\}$ to predict whether it is better to schedule task τ_i as the next task rather than τ_x. With this pairwise classifier, I can determine which single task τ_i is the highest-priority task τ_i^* according to Equation 3.47 by determining which task has the highest cumulative priority in comparison to the other tasks in $\boldsymbol{\tau}$.

In this work, I train a single classifier $f_{priority}(\tau_i, \tau_j)$ to model the behavior of the set of all agents rather than train one $f_{priority}(\tau_i, \tau_j)$ for each agent. $f_{priority}(\tau_i, \tau_j)$ is a function of all features associated with the agents; as such, agents need not be interchangeable, and different sets of features may be associated with each agent.

Next, I must learn to predict whether τ_i^* should be scheduled or the agent should remain idle. Idling is an important characteristic of scheduling with upper and lower-bound deadlines. Thus, I train a second classifier, $f_{act}(\tau_i) \in \{0, 1\}$, that predicts whether or not τ_i should be scheduled. The observations set, O, consists of either examples in which a task was scheduled or those in which no task was scheduled. To train this classifier, I construct a new set of examples according to Equation 3.48, which assigns positive labels to examples from O_m in which a task was scheduled and negative labels to examples in which no task was scheduled.

Finally, I construct a scheduling algorithm to act as an apprentice scheduler (Algorithm 1). This algorithm takes as input the set of tasks $\boldsymbol{\tau}$, agents $\boldsymbol{A}$, temporal constraints (i.e., upper- and lowerbound temporal constraints) relating tasks in the problem $\boldsymbol{TC}$, and the set of task pairs that require the same resources and can therefore not be executed at the same time, $\boldsymbol{\tau_R}$. Lines 1-2 iterate over each agent at each time step. In Line 3, the highest-priority task τ_i^* is determined for a particular agent. In Lines 4-5, τ_i^* is scheduled *if* $f_{act}(\tau_i^*)$ predicts that τ_i^* should be scheduled at the current time. While this approach is greedy, we empirically validate its advantages in Section 3.5, and, in Chapter 4, extend apprenticeship scheduling to efficiently produce globally optimal solutions.

Note that iteration over agents (Line 2) can be performed according to a specified ordering, or one can alternatively learn a more general priority function to select and schedule the best agent-task-resource tuple using $f_{priority}\left(\langle\tau_i, a, r\rangle, \langle\tau_j, a', r'\rangle\right)$, $f_{act}\left(\langle\tau_i, a, r\rangle^*\right)$. In the latter case, the features γ_{τ_i} are mapped to agent-task-resource tuples rather than tasks. I further note that τ_i represents the atomic (i.e., lowest-level) job. For the synthetic evaluation, I use the original formulation $f_{priority}(\tau_i, \tau_j)$. For the ASMD application, I use $f_{priority}\left(\left\langle\tau_i^t, a, r\right\rangle, \left\langle\tau_j^t, a', r'\right\rangle\right)$, where τ_i^t represents the objective of mitigating missile i during time step t, a is the decoy to be deployed, and r is the physical location for that deployment. For the healthcare evaluation, I use $f_{priority}\left(\left\langle\tau_i^j, a, r\right\rangle, \left\langle\tau_p^q, a', r'\right\rangle\right)$, where τ_i^j represents the j^{th} stage of labor for patient i, a is the assigned nurse, and r is the room to which the patient is assigned. For convenience in notation, I refer to this tuple as a "scheduling action."

This hybrid point- and pairwise formulation for predicting which action is best via $f_{priority}\tau_i, \tau_j$, has several key benefits for learning to schedule form expert demonstration. First, one can directly apply standard classification techniques, such as a decision tree, support vector machine, logistic regression, or neural networks. Second, because one only considers two scheduling actions at a time, the model is nonparametric in the number of possible actions. Thus, one can train $f_{priority}(\tau_i, \tau_j)$ on schedules with a agents and n tasks yet apply $f_{priority}(\tau_i, \tau_j)$ to construct a schedule for a problem with a' agents and n' tasks such that $a \neq a'$, $n \neq n'$, and $a * n \neq a' * n'$. Furthermore, one can even train $f_{priority}(\tau_i, \tau_j)$ on demonstrations of a heterogeneous data set of scheduling observations with differing numbers of agents and tasks. Third, the pairwise portion of the formulation provides structure to the learning problem. A formulation that simply concatenated the features of two or more scheduling actions would need to solve the more complex problem of learning the relationships between features and then how to use those relationships to predict the highest priority scheduling action. Such a concatenation approach would suffer from the curse of dimensionality [117]. I note, however, that this strength is also a limitation: I assume that this hybrid model does not lose any information by considering the differences between actions' features. Fourth, the number of positive and negative training examples is balanced given that I simultaneously create one negative label for every positive label. Finally, the model bootstraps the data to create $2 * |\boldsymbol{\tau}|$ examples for each time step, rather than only $|\boldsymbol{\tau}|$ for a purely pointwise model, or even simply 1 example in the case of modeling time series data such as through a Hidden Markov Model.

Note that $f_{act}(\tau_i)$ represents a pointwise formulation for deciding whether or not to take action τ_i. However, one could easily eliminate $f_{act}(\tau_i)$ entirely, folding it into $f_{priority}(\tau_i, \tau_j)$, where τ_j is a null action. The intuition for using a separate function $f_{act}(\tau_i)$ in this work lies in the inherent difference between finding which actions are better than others (i.e., according to an objective function or heuristic) versus evaluating whether that action is feasible (i.e., constraint satisfaction).

Algorithm 1 Pseudocode for an Apprentice Scheduler

ApprenticeScheduler($\boldsymbol{\tau}$,$\boldsymbol{A}$,$\boldsymbol{TC}$,$\boldsymbol{\tau_R}$)

for $t = 0$ to T **do**
    for all agents $a \in \boldsymbol{A}$ **do**
        $\tau_i^* \leftarrow \underset{\tau_i \in \boldsymbol{\tau}}{\operatorname{argmax}} \sum_{\tau_j \in \boldsymbol{\tau}} f_{priority}(\tau_i, \tau_j)$
        if $f_{act}(\tau_i^*) == 1$ **then**
            Schedule τ_i^*
        end if
    end for
end for

3.4 Data Sets

Next, I validate that schedules produced by the learned policies are of comparable quality to those generated by human or synthetic experts. I considered a synthetic data set from the XD [ST-SA-TA] class of problems and a real-world data set from the CD [MT-MA-TA] class of problems defined by Korsah et al. [139].

3.4.1 Synthetic Data Set

For my first investigation, I generated a synthetic data set of scheduling problems in which agents were assigned to complete a set of tasks. Tasks were related through precedence or wait constraints as well as deadline constraints, which could be absolute (relative to the start of the schedule) or relative to another task's start or finish time. Agents were required to access a set of shared resources to execute each task (e.g., the task's physical location). Agents and tasks had defined starting locations, and task locations were static. Each agent traveled at a constant speed between task locations, and agents were only able to perform tasks when present at the corresponding task location. Task completion times were potentially non-uniform and agent-specific, as would be the case for heterogeneous agents. An agent that was incapable of performing a task was assumed to have an infinite completion time for that task. The objective was to minimize the makespan or other time-based performance measures.

This problem definition spans a range of scheduling problems, including the traveling salesman, job-shop scheduling, multi-vehicle routing, and multi-robot task allocation problems, among others. I describe this range as a vehicle routing problem with time windows, temporal dependencies, and resource constraints (VRPTW-TDR), which falls within the XD [ST-SA-TA] class in the taxonomy by [139]: agents perform tasks sequentially (ST), each task requires one agent (SA), and commitments are made over time (TA).

To generate my synthetic data set, I developed a mock scheduling expert that applies one of a set of context-dependent rules based on the composition of the given scheduling problem. This behavior was based upon rules presented in prior work addressing these types of problems [90, 91, 231, 241]. My objective was to show that my apprenticeship scheduling algorithm learns both context-dependent rules as well as to show how to identify the associated context for their correct application.

The mock scheduling expert functions as follows: First, the algorithm collects all alive and enabled tasks $\tau_i \in \boldsymbol{\tau_{AE}}$ as defined by [177]. Consider a pair of tasks τ_i and τ_j, with start and finish times s_i, f_i and s_j, f_j, respectively, such that there is a wait constraint requiring τ_i to start at least $W^{rel}_{\langle \tau_j, \tau_i \rangle}$ units of time after τ_j. A task τ_i is alive and enabled if 1) $t \geq f_j + W^{rel}_{\langle \tau_j, \tau_i \rangle}$ for all such τ_j and $W^{rel}_{\langle \tau_j, \tau_i \rangle}$ in $\boldsymbol{\tau}$ and 2) any absolute wait constraint has been satisfied (i.e., $W^{abs}_{\tau_i} \leq t$).

Next, the heuristic iterates over each agent to identify the highest-priority task, τ_i^*, to schedule for that agent. The algorithm determines which scheduling rule is most appropriate to apply for each agent. If agent speed is sufficiently slow ($\leq$ 1 m/s), travel time will become the major bottleneck. If agents move quickly but heavily utilize one or more resources R ($\sum_{\tau_i} \sum_{\tau_j} 1_{R_{\tau_i} = R_{\tau_j}} \geq$ c for some constant c), use of these resources can become the bottleneck. Otherwise, task durations and associated wait constraints are generally most important.

If the algorithm identifies travel distance as the primary bottleneck, it chooses the next task by applying a priority rule well-suited for vehicle routing that minimizes a weighted, linear combination of features [84, 231] comprised of the distance and angle relative to the origin between agent a and τ_j. This rule is depicted in Equation 3.49, where $\vec{l_x}$ is the location of τ_j, $\vec{l_a}$ is the location of agent a, θ_{xa} is the relative angle between the vector from origin to the agent location and the origin to the location of

τ_j, and α_1 and α_2 are weighting constants.

$$\tau_i^* \leftarrow \underset{\tau_j \in \boldsymbol{\tau_{AE}}}{\operatorname{argmin}} \left(\|\vec{l}_x - \vec{l}_a\| + \alpha_1 \theta_{xa} + \alpha_2 \|\vec{l}_x - \vec{l}_a\| \theta_{xa} \right) \tag{3.49}$$

If the algorithm identifies resource contention as the most important bottleneck, it employs a rule to mitigate resource-contention in multi-robot, multi-resource problems based on prior work in scheduling for multi-robot teams [91]. Specifically, the algorithm uses Equation 3.50 to select the high-priority task to schedule next, where d_{τ_j} is the deadline of τ_j and α_3 is a weighting constant.

$$\tau_i^* \leftarrow \underset{\tau_j \in \boldsymbol{\tau_{AE}}}{\operatorname{argmax}} \left(\left(\sum_{\tau_i} \sum_{\tau_j} 1_{R_{\tau_i} = R_{\tau_j}} \right) - \alpha_3 d_{\tau_j} \right) \tag{3.50}$$

If the algorithm decides that temporal requirements are the major bottleneck, it employs an Earliest Deadline First rule (Equation 3.51), which performs well across many scheduling domains [43, 90, 91].

$$\tau_i^* \leftarrow \underset{\tau_j \in \boldsymbol{\tau_{AE}}}{\operatorname{argmin}} d_{\tau_j} \tag{3.51}$$

After selecting the most important task, τ_i^*, the algorithm determines whether the resource required for τ_i^*, $R_{\tau_i^*}$, is idle and whether the agent is able to travel to the task location by time t. If these constraints are satisfied, the heuristic schedules task τ_i^* at time t. (An agent is able to reach task τ_i^* if $t \geq f_j + k\left(l_i - l_j\right)/\|l_i - l_j\|$ for all $\tau_j \in \boldsymbol{\tau}$ that the agent has already completed, where k is the agent's speed.)

I constructed the synthetic data set for two homogeneous agents and 20 partially ordered tasks located within a 20 x 20 grid.

Algorithmic Description

For reproducibility, I provide the pseudocode of the mock expert shown in Figure 3-3. In Line 1, the heuristic retrieves all alive and enabled tasks $\tau_i \in \boldsymbol{\tau_{AE}}$. A task τ_i is alive and enabled if all of its wait constraints have been satisfied (i.e.,

$t \geq f_{\tau_j} + W^{rel}_{\langle \tau_j, \tau_i \rangle}, \forall W^{rel}_{\langle \tau_j, \tau_i \rangle})$. Next, the heuristic iterates over each agent and each task to find the highest priority task τ_i^* to schedule for each agent a. In Lines 3-12, the algorithm determines which heuristic is most appropriate to apply.

If the speed of the agents is sufficiently slow, then the travel distance will become the major bottleneck. If the agents are fast, but there are one or more resources that are heavily utilized, then these resources can become the bottleneck. Otherwise, the duration of the task and their associated wait constraints are generally the most important to consider.

In Line 3, the algorithm decides travel distance as the most important bottleneck. As such, the algorithm applies a heuristic rule to find the task that maximizes a weighted, linear combination of hand-crafted features comprised of the distance and angle relative to the origin between agent a and τ_i as well as the distance, the angle relative to the origin between agent τ_j and agent b, and an indicator term for whether τ_i must be executed to satisfy a wait constraint for another task τ_j. Here, l_{τ_i}, l_a, and l_b are the locations in $\mathbb{R}^2$ of task τ_i, agent a, and agent $b \neq a$.

In Line 5, the algorithm determines that there may be a resource bottleneck and tries to alleviate this potential bottleneck. As such, the algorithm applies a heuristic rule that returns the task $\tau_i^* \in \boldsymbol{\tau_{AE}}$ that maximizes a weighted, linear combination of the commonality of the task's required resource and its deadline. Lastly, if neither travel distance or resource contention are perceived to be the major bottlenecks, the algorithm applies an Earliest Deadline First rule.

3.4.2 Real-World Data Set: ASMD

A real-world data set was collected, consisting of human demonstrators of various skill levels solving the ASMD problem. Data was collected from domain experts playing a serious game, called Strike Group Defender[2] (SGD), for ASMD training. Game scenarios involved five types of decoys and ten types of threats. The threats were randomly generated for each played scenario, thereby promoting the development of strategies that were robust to a varied distribution of threat scenarios. Each decoy

[2]SGD was developed by Pipeworks Studio in Eugene, Oregon, USA.

MockHeuristic(τ**, A, TC,** $\tau_{\mathbf{R}}$**, AC)**

$\boldsymbol{\tau_{AE}} \leftarrow$ all alive and enabled $\tau_i \in \boldsymbol{\tau}$
for all agents $a \in A$ **do**
  if $speed \leq 1\ m\backslash s$ **then**
    $\tau_i^* = \underset{\tau_i \in \boldsymbol{\tau_{AE}}}{\operatorname{argmin}} \left(\alpha_1 \| l_{\tau_i} - l_a \| + \alpha_2 \frac{acos(l_{\tau_i} \cdot l_a)}{\|l_{\tau_i}\| \|l_a\|} + \alpha_3 \| l_b - l_a \| + \alpha_4 \frac{acos(l_b \cdot l_{\tau_i})}{\|l_b\| \|l_{\tau_i}\|} + 1^{rel}_{\left(\exists W_{\langle \tau_i, \tau_j \rangle}\right)} \right)$
  else if $\sum_{\tau_i \in \boldsymbol{\tau}} \sum_{\tau_x^y \in \boldsymbol{\tau}} 1_{R_{\tau_i} = R_{\tau_j}} \geq \epsilon$ **then**
    $\tau_i^* = \underset{\tau_i \in \boldsymbol{\tau_{AE}}}{\operatorname{argmax}} \left(\alpha_1' \left(\sum_{\tau_j} 1_{\left(R_{\tau_i} = R_{\tau_j}\right)} \right) + \alpha_2'' \left(\max\left(d_{\tau_j}\right) - d_{\tau_i} \right) \right)$
  else
    $\tau_i^* = \underset{\tau_i \in \boldsymbol{\tau_{AE}}}{\operatorname{argmin}} \left(d_{\tau_i} \right)$
  end if
  if a and r can schedule τ_i^* at time t **then**
    schedule τ_i^*
  end if
end for

Figure 3-3: Pseudocode for the Mock Heuristic.

had a specified effectiveness against each threat type. Players attempted to deploy a set of decoys using the correct types at the correct locations and times in order to distract incoming missiles. Threats were launched over time; an effective deployment at time t could become counterproductive in the future as new enemy missiles were launched.

Games were scored as follows: $10,000$ points were received each time a threat was neutralized and 2 points were received for each second a threat spent homing in on a decoy. $5,000$ points were deducted for each threat impact and 1 point was deducted for each second a threat spent homing in the players ship. 25-$1,000$ points were subtracted for each decoy deployment, with the deducted point value depending upon the decoy type.

The collected data set consisted of 311 games played by 35 humans across 45 threat configurations, or "levels." From this set, I also separately analyzed 16 threat configurations such that each configuration included at least one human demonstration in which the ship was protected from all enemy missiles. For these 16 threat configurations, there were 162 total games played by 27 unique human demonstrators. The sample population consisted of technical fellows and associates, as well as contractors at a federally funded research and development center (FFDRC), and their expertise varied from "generally knowledgeable about the ASMD problem" to "domain experts" with professional experience or training in ASMD.

3.4.3 Real World Dataset: Labor and Delivery

I collected another real-world data set – this time consisting of resource nurses on the labor and delivery floor at Beth Israel Deaconess Medical Center. To collect data of decisions from resource nurses, a high-fidelity simulation of a labor and delivery floor was developed, as shown in Figure 3-4. I spearheaded the development of this simulation under the auspices of a hospital quality improvement project as a training tool over a year-long, rigorous design and iteration process that included workshops with nurses, physicians, and medical students to ensure the tool accurately captured the role of a resource nurse. Parameters within the simulation (e.g., arrival of patients, timelines on progression through labor) were drawn from medical textbooks and papers and modified through alpha and beta testing to ensure that the simulation closely mirrored the patient population and nurse experience at my partner hospital.

I invited expert resource nurses to play this simulation to collect a dataset for training the apprenticeship scheduling algorithm. This dataset was generated by seven resource nurses working with the simulation for a total of $2^1/_2$ hours, simulating 60 hours of elapsed time on a real labor floor. This yielded a dataset of more than $3,013$ individual decisions.

3.5 Empirical Evaluation

In this section, I evaluate my prototype for apprenticeship scheduling on the synthetic and real-world data sets.

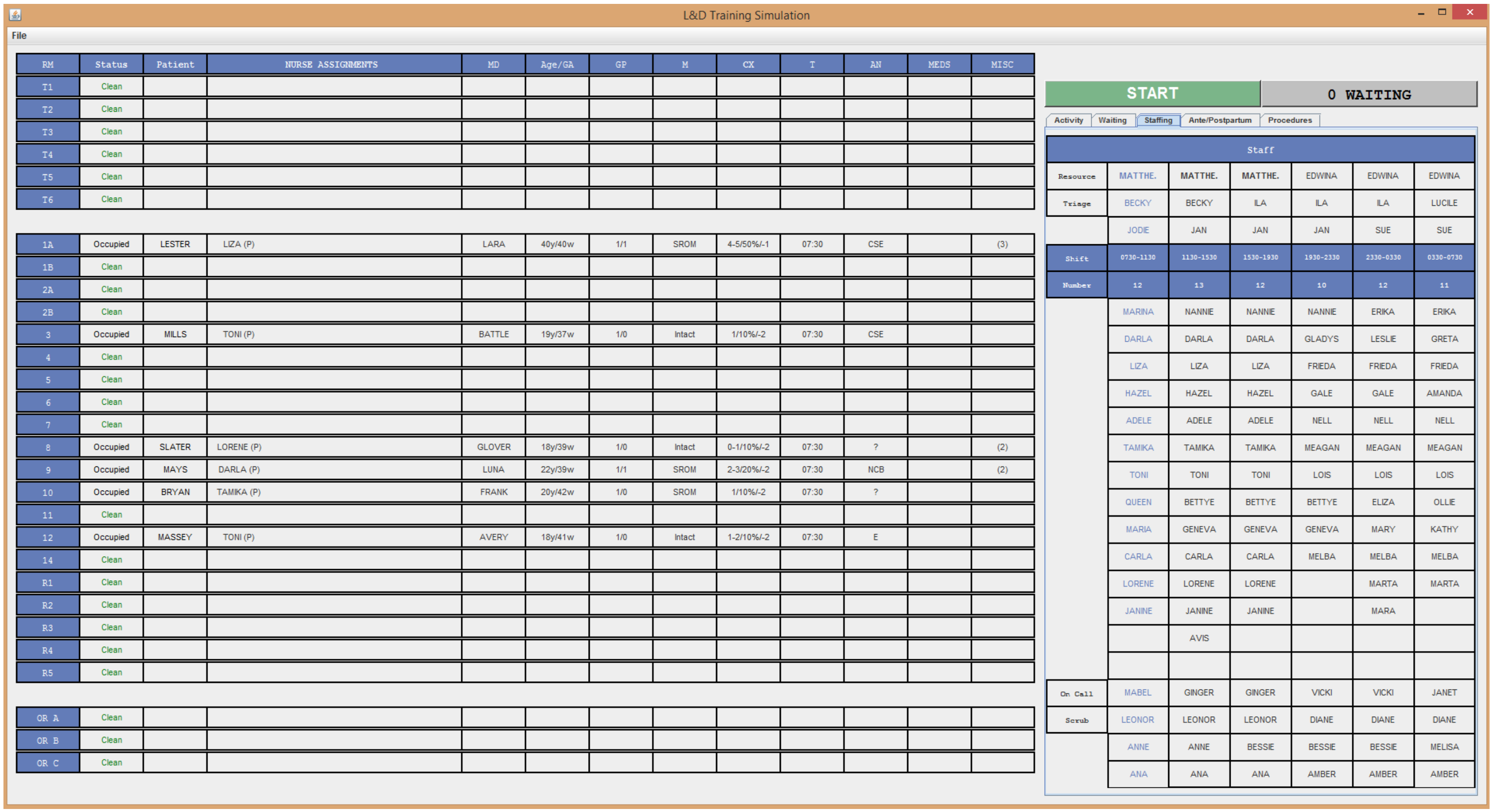

Figure 3-4: Screen capture of the simulation used in to capture data for the nursing data set.

3.5.1 Synthetic Data Set

I trained my model using a decision tree, KNN classifier, logistic regression (logit) model, support vector machine with a radial basis function kernel (SVM-RBF), and a neural network to learn $f_{priority}(.,.)$ and $f_{act}(.)$. I randomly sampled 85% of the data for training and 15% for testing.

I defined the features as follows: The high-level feature vector of the task set, ξ_{τ}, was comprised of the agents' speed and the degree of resource contention $\sum_{\tau_i} \sum_{\tau_j} 1_{R_{\tau_i} = R_{\tau_j}}$. The task-specific feature vector γ_{τ_i} was comprised of the task's deadline, a binary indicator for whether or not the task's precedence constraints had been satisfied, the number of other tasks sharing the given task's resource, a binary indicator for whether or not the given task's resource was available, the travel time remaining to reach the task location, the distance agent a would travel to reach τ_i, and the angular difference between the vector describing the location of agent a and the vector describing the position of τ_i relative to agent a.

I compared the performance of my hybrid pairwise-pointwise approach with a purely pointwise approach as well as a naïve approach. In the pointwise approach, training examples for selecting the highest-priority task were of the form ${}^{rank}\phi^m_{\tau_i} := [\xi_{\boldsymbol{\tau}}, \gamma_{\tau_i}]$. The label $\gamma^m_{\tau_i}$ was equal to 1 if task τ_i was scheduled in observation m, and was 0 otherwise. In the naïve approach, examples were comprised of an input vector that concatenated the high-level features of the task set and the task-specific features of the form ${}^{rank}\phi^m := [\xi_{\boldsymbol{\tau}}, \gamma_{\tau_1}, \gamma_{\tau_2}, \ldots, \gamma_{\tau_n}]$; labels y^m were given by the index of the task τ_i scheduled in observation m.

Figures 3-5 and 3-6 depict the sensitivity (true positive rate) and specificity (true negative rate) of the model, respectively. I found that a pairwise model outperformed the pointwise and naïve approaches. Within the pairwise model, a decision tree yielded the best performance. The trained decision tree was able to identify the correct task and when to schedule that task 95% of the time, and was able to accurately predict when no task should be scheduled 96% of the time.

To more fully understand the performance of a decision tree trained with a pairwise

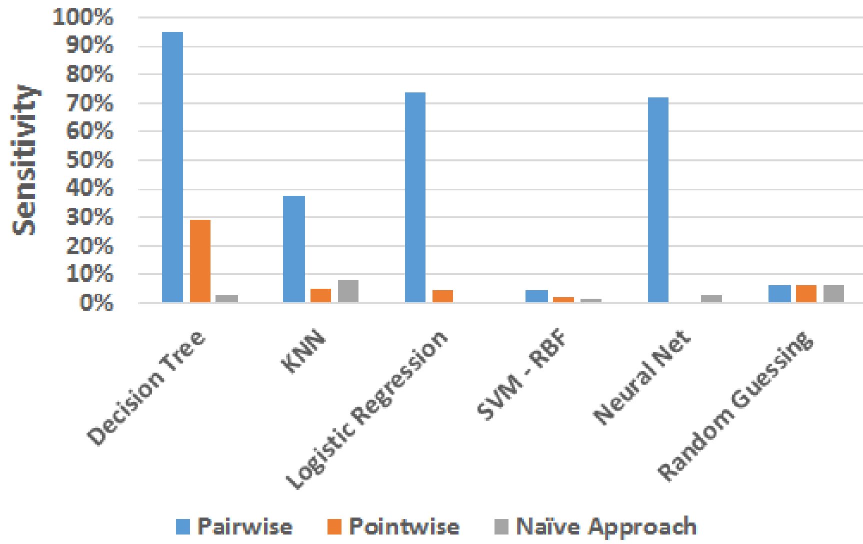

Figure 3-5: Sensitivity of machine learning techniques using the pairwise, pointwise, and naïve approaches.

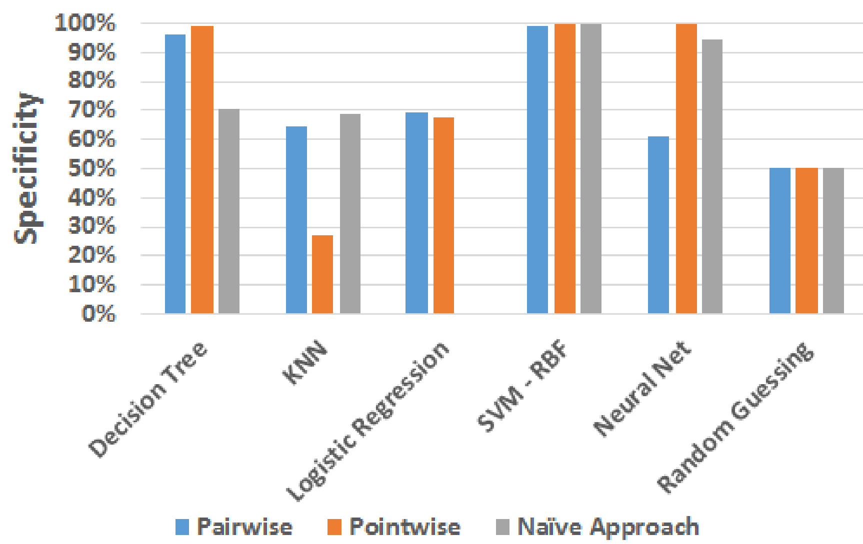

Figure 3-6: Specificity of machine learning techniques using the pairwise, pointwise, and naïve approaches.

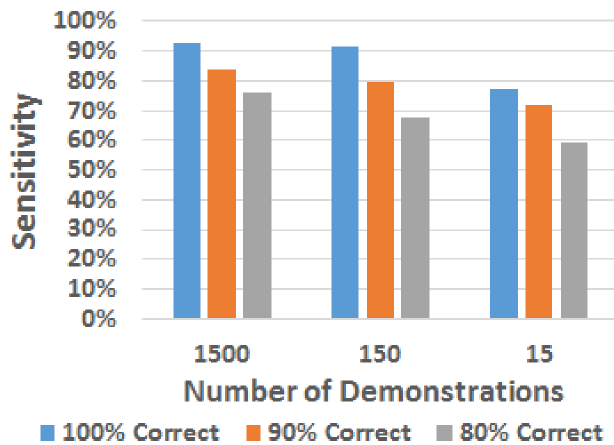

Figure 3-7: Sensitivity for a pairwise decision tree varying the number and proportion of correct demonstrations.

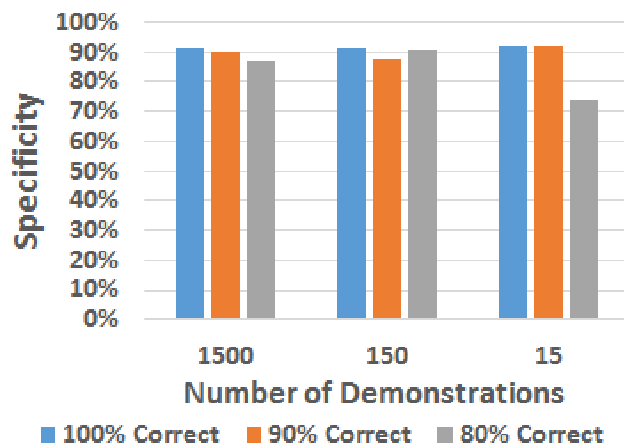

Figure 3-8: Specificity for a pairwise decision tree varying the number and proportion of correct demonstrations.

model as a function of the number and quality of training examples, I trained decision trees with my pairwise model using 15, 150, and 1,500 demonstrations. The sensitivity and specificity depicted in Figures 3-7 and 3-8 for 15 and 150 demonstrations are the mean sensitivity and specificity of 10 models trained via random sub-sampling without replacement. I also varied the quality of the training examples, assuming the demonstrator was operating under an ϵ-greedy approach with a $(1 - \epsilon)$ probability of selecting the correct task to schedule and selecting another task from a uniform distribution otherwise. This assumption is conservative; a demonstrator making an error would be more likely to pick the second- or third-best task than to select a task at random.

Training a model based on pairwise comparison between the scheduled task and unscheduled tasks effectively produced a comparable policy to that of the synthetic expert. The decision tree model performed well due to the modal nature of the multifaceted scheduling heuristic. Note that this dataset was composed of scheduling strategies with mixed discrete-continuous functional components; performance could potentially be improved upon in future work by combining decision trees with logistic regression. This hybrid learning approach has been successful in prior machine learning classification tasks [143] and can be readily applied to this apprenticeship scheduling framework. There is also an opportunity to improve performance through hyper-parameter tuning (e.g., to select the minimum number of examples in each leaf of the decision tree). Comprehensive investigation of the relative benefits for a range of learning techniques is left for future work.

Importantly, I note that the results presented in Figures 3-5-3-8 were achieved without any hyper-parameter tuning. For example, with the decision tree, I did not perform an inner cross-validation loop to estimate the minimum number of examples in each leaf to achieve the best performance. The purpose of this analysis was to merely show that, with our pair-wise approach, we can in fact accurately learn expert heuristics from example. In the following section, I investigate how apprenticeship scheduling using a decision tree classifier can be improved via an inner cross validation loop to tune the model's hyper-parameters.

Additional Evaluation

My initial analysis above was performed to identify which techniques have inherent advantages that can be realized without extensive hyper-parameter tuning. This initial analysis showed that the pairwise formulation for apprenticeship scheduling, in conjunction with a decision tree classifier, has advantages over alternative formulations for learning a high-quality scheduling policy. Given evidence of this advantage, I now further evaluate the potential of the pairwise formulation with hyper-parameter tuning.

To improve the performance of the model, I manipulated the "leafiness" of the decision tree to find the best setting to increase the accuracy of the apprenticeship scheduler. Specifically, I varied the minimum number of training examples required in each leaf of the decision tree. As the minimum number of examples required for each leaf decreases, one increases the chance of over-fitting to the data. Conversely, as the minimum number increases, one increases the chance of not learning a helpful policy (i.e., under-fitting). To identify the best number of leaves for generalization, I tested values for the minimum number of examples required for each leaf of the decision tree in the set $\{1, 5, 10, 25, 50, 100, 250, 500, 1000\}$. If the minimum number of examples in each leaf exceeded the total number of examples, the setting was trivially set to the total number of examples available for training.

I performed 5-fold cross-validation for each value of examples as follows. I trained an apprentice scheduler on four-fifths of the training data and test on one-fifth of the data. We record the average testing accuracy across each of the five folds. Then, the setting of the minimum number of examples required for each leaf that yielded the best accuracy during 5-fold cross validation was used to train a full apprenticeship scheduling model on all of the training data, which was 85% of the total data. Finally, the full apprenticeship scheduling model was tested on the 15% of the total data reserved for testing. Thus, none of the data used for testing the full model was used to estimate the best setting for the leafiness of the tree. Finally, I repeat this procedure ten times, randomly subsampling the data and taking the average performance across

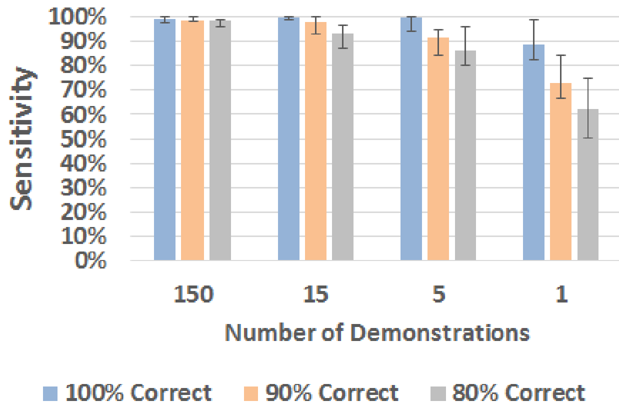

Figure 3-9: Sensitivity for a pairwise decision tree, tuned for leafiness, varying the number and proportion of correct demonstrations.

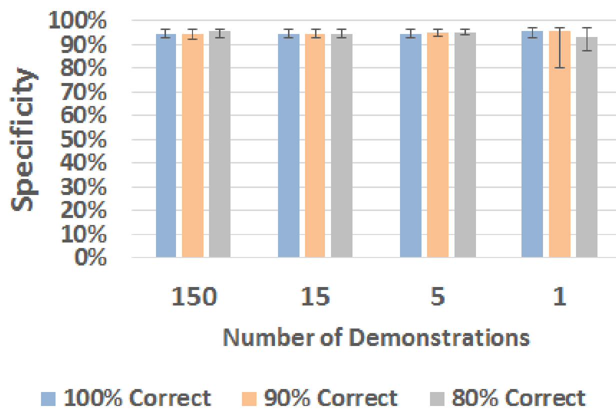

Figure 3-10: Specificity for a pairwise decision tree, tuned for leafiness, varying the number and proportion of correct demonstrations with homogeneous agents.

the ten trials.

The sensitivity and specificity of the fully-trained apprenticeship scheduling algorithm are depicted in Figures 3-11 and 3-12 for 1, 5, 15, and 150 scheduling demonstrations. As before, I also varied the quality of the training examples, assuming the demonstrator was operating under an ϵ-greedy approach with a $(1-\epsilon)$ probability of selecting the correct task to schedule and selecting another task from a uniform distribution otherwise.

Furthermore, to make the problem more complex, I also constructed an entirely new data set of examples from the mock heuristic in which the agents are now heterogeneous, meaning that task completion times are agent specific. Each agent's task completion time, C_i^a, was drawn from a Gaussian distribution with a task-specific

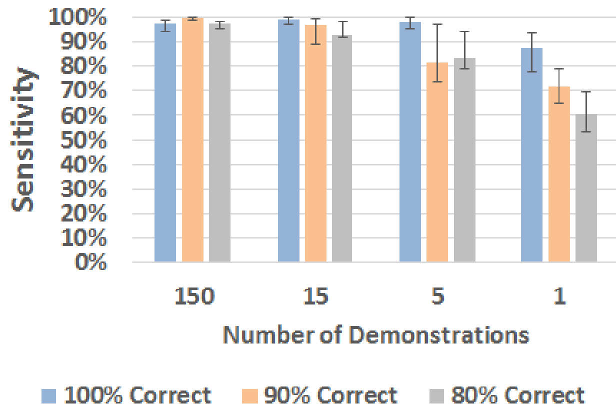

Figure 3-11: Sensitivity for a pairwise decision tree, tuned for leafiness, varying the number and proportion of correct demonstrations. The corresponding data set was comprised of schedules with heterogeneous agents.

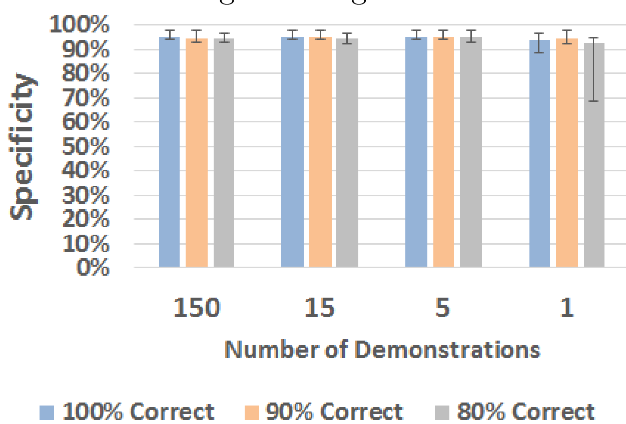

Figure 3-12: Specificity for a pairwise decision tree, tuned for leafiness, varying the number and proportion of correct demonstrations. The corresponding data set was comprised of schedules with heterogeneous agents.

mean μ_i and common variance σ_i. Values were limited to the range $[1, 10]$. The sensitivity and specificity of the fully-trained apprenticeship scheduling algorithm on the heterogeneous data set are likewise depicted in Figures 3-9 and 3-10

For both the homogeneous and heterogeneous cases, I find that the apprenticeship scheduling algorithm was able to average $\geq 90\%$ sensitivity and specificity either with five perfect schedules or fifteen schedules generated by an operating making mistakes 20% of the time. Hyper-parameter tuning was able to substantially increase the sensitivity of the model from 59% to 82% for five scheduling examples generated by an operating making mistakes 20% of the time. Recall that a schedule consists of allocating twenty tasks to two workers and sequencing those tasks in time.

3.5.2 Real-World Data Set: ASMD

I trained and tested a decision tree on my pairwise scheduling model via leave-one-out cross-validation using 16 real demonstrations in which a player successfully protected the ship from all enemy missiles. Each demonstration originated from a unique threat scenario. Features for each decoy/missile pair (or null decoy deployment due to inaction) included indicators for whether a decoy had been placed such that a missile was successfully distracted by that decoy, whether a missile would be lured into hitting the ship due to decoy placement, or whether a missile would be unaffected by decoy placement.

Across all 16 scenarios, the mean player score was $74,728 \pm 26,824$. With only 15 examples of expert human demonstrations, my apprenticeship scheduling model achieved a mean score of $87,540$ with a standard deviation of $16,842$.

I performed a statistical analysis to evaluate my hypothesis that the scores produced by the learned policy would be statistically significantly better than the scores achieved by the human demonstrators. The null hypothesis stated that the number of scenarios in which the apprenticeship scheduling model achieved superior performance would be less than or equal to the number of scenarios in which the mean score of the human demonstrators was superior to that of the apprenticeship scheduler. I set the significance level at $\alpha = 0.05$, which means that the risk of identifying a difference between the mean scores earned by the apprenticeship scheduler and the set of human performers when no such difference exists is less than 5%.

Results from a binomial test rejected the null hypothesis, indicating that the learned scheduling policy performed better than the human demonstrators in significantly more scenarios (12 versus 4 scenarios; $p < 0.011$). In other words, I can say with 95% certainty that the apprenticeship scheduler is superior to the average human demonstrator. This promising result was achieved using a relatively small training set, and suggests that learned policy can form the basis for a training tool to improve the average player's score.

3.5.3 Real-World Data Set: Labor and Delivery

To test the efficacy of the apprenticeship scheduling algorithm for labor and delivery, I asked a group of physicians and registered nurses (one man and sixteen women) to validate whether the apprenticeship scheduler generated advice in-keeping with what a resource nurse would do. This group consisted of seventeen physicians and registered nurses participated in the experiment. The participants were recruited from the partner hospital's obstetrics department via email and word-of-mouth. I found that the apprenticeship scheduling algorithm, trained on a data set of expert resource nurses, produced high-quality recommendations accepted by nurses and physicians at a compliance rate of 90%. This indicates that an apprentice scheduler may be able to learn context-specific decision strategies and apply them to make reasonable suggestions for which tasks to perform and when. This initial finding in labor and delivery is promising, and, in Chapter 5, I build on this finding to investigate the development of a full robotic system based on apprenticeship scheduling.

3.6 Anomalies and Future Work

The core of my apprenticeship scheduling algorithm is comprised of learning a classifier, $f_{priority}(\tau_i, \tau_j)$, to predict whether a human expert would take action τ_i over τ_j. The output of $f_{priority}(\tau_i, \tau_j)$ is a probability in $[0, 1]$. This pairwise approach has a number of key advantages. For example, it is nonparametric in the number of tasks, meaning one can learn from problems with n actions and apply that knowledge to problems with $n' \neq n$ actions. However, there are two interesting anomalies inherent in this approach. First, one could hypothetically evaluate $f_{priority}(\tau_i, \tau_j)$ and find that it predicts that the expert has a higher probability of taking action τ_i than τ_j; yet, evaluating $\underset{\tau_i \in \boldsymbol{\tau}}{\operatorname{argmax}} \sum_{\tau_j \in \boldsymbol{\tau}} f_{priority}(\tau_i, \tau_j)$ could predict that τ_j is the action most likely to be taken by the expert. The second anomaly entails the lack of a guarantee that the transitive property will hold for arbitrary $f_{priority}(\tau_i, \tau_j)$. For example, it could be that $f_{priority}(\tau_i, \tau_j) > 0.5$, $f_{priority}(\tau_j, \tau_k) > 0.5$, but $f_{priority}(\tau_k, \tau_i) > 0.5$ for some τ_i, τ_j, and τ_k. It is unclear how this property would affect the macroscopic behavior of the algorithm, but it is worth investigating in future work. Through my evaluation, I have shown that my formulation for apprenticeship scheduling can learn high quality policies from human domain experts' demonstrations. However, an interesting area of future work would be to study these anomalies, quantify their effects – if any – and develop a formulation to alter their effects. For example, one could consider the following formulation in Equations 3.52 through 3.57 when learning a decision tree model, T^*, for apprenticeship scheduling.

$$T^* = \underset{T}{\operatorname{argmin}}\, \mathbb{E}_{\theta,y}[L(y^m_{\langle\tau_i,\tau_j\rangle}, T({}^{rank}\theta^m_{\langle\tau_i,\tau_j\rangle}))] \tag{3.52}$$

subject to

$$T({}^{rank}\theta^m_{\langle\tau_i,\tau_j\rangle}) > 0.5 + M(1 - Z_{i,j}), \forall \tau_i, \tau_j \tag{3.53}$$

$$T({}^{rank}\theta^m_{\langle\tau_i,\tau_j\rangle}) < 0.5 + M(Z_{i,j}), \forall \tau_i, \tau_j \tag{3.54}$$

$$\sum_{\tau_k \in \boldsymbol{\tau}} T({}^{rank}\theta^m_{\langle\tau_i,\tau_k\rangle}) - \sum_{\tau_k \in \boldsymbol{\tau}} T({}^{rank}\theta^m_{\langle\tau_j,\tau_k\rangle}) > M(1 - Z_{i,j}), \forall \tau_i, \tau_j \tag{3.55}$$

$$\sum_{\tau_k \in \boldsymbol{\tau}} T({}^{rank}\theta^m_{\langle\tau_i,\tau_k\rangle}) - \sum_{\tau_k \in \boldsymbol{\tau}} T({}^{rank}\theta^m_{\langle\tau_j,\tau_k\rangle}) < M(Z_{i,j}), \forall \tau_i, \tau_j \tag{3.56}$$

$$Z_{i,j} + Z_{j,k} - 1 \geq Z_{i,k}, \forall \tau_i, \tau_j, \tau_k \tag{3.57}$$

Equation 3.52 states that we want to find the decision tree T^* amongst all possible trees T that minimizes an expected loss function, L. Recall from Section 3.3 that $y^m_{\langle\tau_i,\tau_j\rangle}$ is the binary label given to an observation to indicate whether the human demonstrator took action τ_i or τ_j. Further, ${}^{rank}\theta^m_{\langle\tau_i,\tau_j\rangle}$ is the corresponding feature vector from that observation. Equations 3.53 through 3.56 force the pairwise comparisons to agree with the cumulative ranking. $Z_{i,j}$ is a binary decision variable that is equal to one when τ_i is expected to be taken over τ_j and zero when τ_j is expected to be taken over τ_i. Finally, M is a large, positive number. Finally, Equation 3.57 requires that 3.57 the transitive property holds for T. Specifically, if τ_i is predicted to be more likely than τ_j (i.e, $Z_{i,j} = 1$), and τ_j (i.e, $Z_{j,k} = 1$) is more likely than τ_k, then τ_i should also be predicted to be more likely than τ_k (i.e, $Z_{i,k} = 1$).

3.7 Conclusions

In this chapter, I propose a technique for apprenticeship scheduling that relies on a pairwise comparison of scheduled and unscheduled tasks to learn a model for task prioritization. I validated my apprenticeship scheduling algorithm using both a synthetic data set covering a variety of scheduling problems with lower- and upperbound temporal constraints, resource constraints, and travel distance considerations, as well as a two real-world data sets in which human demonstrators solved a variant of the weapon-to-target assignment problem and a resource allocation problem in healthcare. My approach was able to learn high-quality policies for decision-making in labor and delivery as well as anti-ship missile defense. In the next chapter, I build upon this work in policy learning to build a machine learning-optimization framework to learn to produce super-human scheduling solutions.

Chapter 4

Learning to Make Super-Human Scheduling Decisions

4.1 Introduction

In Chapter 3, I developed a method, apprenticeship scheduling, to learn to schedule from human expert demonstration. This technique is the first step in a larger picture. Apprenticeship scheduling is a form of policy learning specifically suited for scheduling. However, the quality of the solutions is limited by the quantity and quality of the humans' demonstrations. Further, employing policy learning to sequentially construct a schedule has drawbacks. Small deviations from the ideal scheduling action at each step may result in a large, cumulative deviation from the intended, optimal schedule. The next logical step is then to combine policy learning with a goal-based feedback mechanism to improve quality of the schedules produced by the apprenticeship scheduling algorithm.

In this chapter, I propose Collaborative Optimization via Apprenticeship Scheduling (COVAS), an approach that incorporates machine learning from human expert demonstration (i.e., apprenticeship scheduling), in conjunction with optimization, to automatically and efficiently produce optimal solutions to challenging real-world scheduling problems. My method use apprenticeship scheduling to perform policy learning using a training dataset comprised of schedules demonstrated by humans [89].

COVAS uses apprenticeship scheduling to generate a favorable (if suboptimal) initial solution to a new scheduling problem. COVAS uses this initial solution to provide a tight bound on the value of the optimal solution, substantially improving the efficiency of a branch-and-bound search for an optimal schedule. To ensure compatibility between the apprentice scheduler's solution and the MILP formulation, I augment the apprenticeship scheduler to solve a constraint satisfaction problem, ensuring that the execution of each scheduling commitment does not directly result in infeasibility.

I demonstrate my approach by solving a real-world anti-ship missile defense problem, and report that COVAS produces substantially superior solutions to those produced by human domain experts, at a rate 9.5 times faster than an optimization approach that does not incorporate human expert demonstration.

4.2 Model for Collaborative Optimization via Apprenticeship Scheduling

Here, I provide an overview of the COVAS architecture, and then present its two components: the policy learning and optimization routines.

4.2.1 COVAS Architecture

The system (Figure 4-1) takes as input a set of domain expert scheduling demonstrations (e.g., Gantt charts) that contains information describing which agents complete which tasks, when, and where. These demonstrations are passed to my apprenticeship scheduling algorithm that learns a classifier, $f_{priority}(\tau_i, \tau_j)$, to predict whether the demonstrator(s) would have chosen scheduling action τ_i over action $\tau_j \in \boldsymbol{\tau}$.

Recall from Section 3.3 that, for modeling the ASMD problem domain, τ_i represents a task-agent-resource tuple, $\langle \tau_i^t, a, r \rangle$, where τ_i^t represents the objective (i.e., high-level task) of mitigating missile i during time step t, a is the decoy (i.e., agent) to be deployed, and r is the physical location for that deployment (i.e., where the task will be performed). For convenience in notation, I simply refer to this tuple as

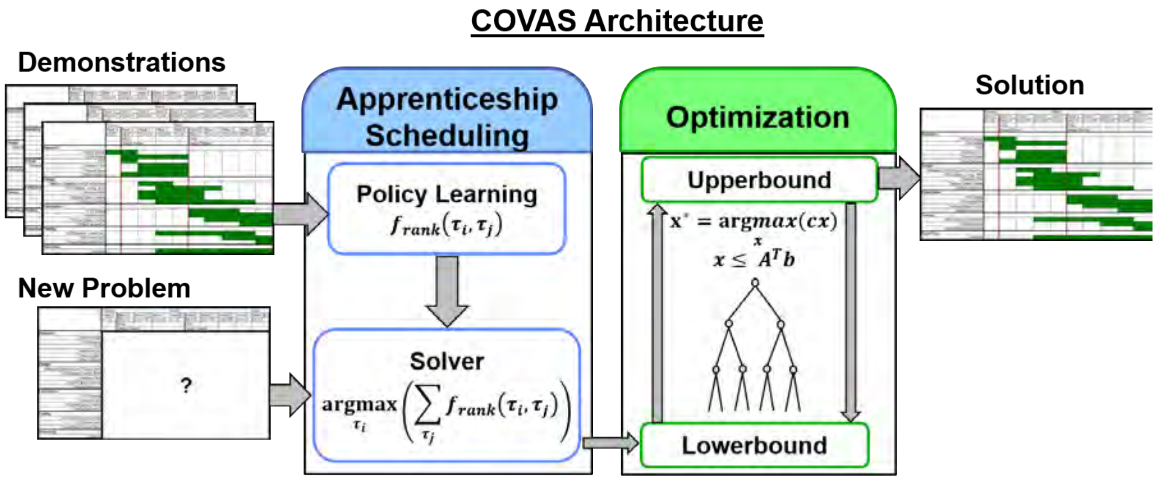

Figure 4-1: The COVAS architecture.

"action τ_i."

Next, COVAS uses $f_{priority}(\tau_i, \tau_j)$ to construct a schedule for a new problem. COVAS creates an event-based simulation of this new problem and runs the simulation in time until all tasks have been completed. In order to complete tasks, COVAS uses $f_{priority}(\tau_i, \tau_j)$ at each moment in time to select the best scheduling action to take. I describe this process in detail in the next section.

Next, COVAS provides this output as an initial seed solution to an optimization subroutine (i.e., a MILP solver). The initial solution produced by the apprenticeship scheduler improves the efficiency of a search by providing an empirical lowerbound on the objective function value of the optimal schedule. An upperbound can be simultaneously obtained by solving an LP-relaxation of the MILP formulation. These bounds are then used to inform a branch-and-bound search over the integer variables [22], enabling the search algorithm to prune areas of the search tree and focus its search on areas that can yield the optimal solution. After the algorithm has identified an upper- and lowerbound within some threshold, COVAS returns the solutions that have been proven optimal within that threshold. Thus, an operator can use COVAS as an anytime algorithm and terminate the optimization upon finding a solution that is acceptable within a provable bound.

4.2.2 Apprenticeship Scheduling Subroutine

In Chapter 3, I developed the theory for the apprenticeship scheduling algorithm. The algorithm is centered around learning a classifier, $f_{priority}(\tau_i, \tau_j)$, to predict whether an expert would take scheduling action τ_i over τ_j. With this function, we can then predict which single action, τ_i^*, amongst a set of actions $\boldsymbol{\tau}$, the expert would take by applying Equation 4.1.

$$\tau_i^* = \underset{\tau_i \in \boldsymbol{\tau}}{\operatorname{argmax}} \sum_{\tau_x \in \boldsymbol{\tau}} f_{priority}(\tau_i, \tau_x) \tag{4.1}$$

In this chapter, I build upon and integrate this formulation into my collaborative-optimization via apprenticeship scheduling framework.

As a subroutine within COVAS, $f_{priority}(\tau_i, \tau_j)$ is applied to obtain the initial solution to a new scheduling problem as follows: First, the user must instantiate a simulation of the scheduling domain; then, at each time step in the simulation, take the scheduling action predicted by Equation 4.1 to be the action that the human demonstrators would take. This equation identifies the task τ_i with the highest importance marginalized over all other tasks $\tau_j \in \boldsymbol{\tau}$. If Equation 4.1 predicts an action should be taken, that action is taken and the equation is re-evaluated. Actions continue to be taken until Equation 4.1 predicts that a null action should be executed.

Different from Chapter 3, each selected action is validated using a schedulability test (i.e., solving a constraint satisfaction problem) to ensure that direct application of that action does not violate the constraints of the new problem. For example, in anti-ship missile defense, one would check to ensure that action does not result in a suicidal deployment (i.e., the decoy directly causes a missile to impact the ship). The test must be designed to be fast so as to make the benefit to feasibility and optimality in the resulting schedule worth the additional complexity. If, at a given time step, τ_i^* does not pass the schedulability test, COVAS uses Equation 4.1 for all $\tau_i \in \boldsymbol{\tau} \backslash \tau_i^*$ to consider the second-best action. If no action passes the schedulability test, no action is taken during that time step.

While the schedulability test forces the apprenticeship scheduling algorithm to follow a subset of the full constraints in the MILP formulation, it is possible that the algorithm may not successfully complete all tasks. Here, I model tasks as optional and use the objective function to maximize the total number of tasks completed. In turn, constraints for a task that the apprenticeship scheduling algorithm did not satisfactorily complete can be turned off, with a corresponding penalty in the objective function score. Thus, an initial seed solution that has not completed all tasks (i.e., satisfied all constraints to complete the task) can still be helpful for seeding the MILP.

4.2.3 Optimization Subroutine

For optimization, I employ mathematical programming techniques to solve mixed-integer linear programs via branch-and-bound search. COVAS incorporates the solution produced by the apprenticeship scheduler to seed a mathematical programming solver with an initial solution. This is a built-in capability provided by many off-the-shelf, state-of-the-art MILP solvers, including CPLEX[1] and Gurobi[2]. This seed provides a tight bound on the value of the optimal solution, which serves to dramatically cut the search space, allowing the system to more quickly hone in on the area containing the optimal solution and, in turn, more quickly solve the optimization problem. Furthermore, this approach allows COVAS to quickly achieve a bound on the optimality of the solution provided by the apprenticeship scheduling subroutine. In such a manner, an operator can determine whether the apprenticeship scheduling solution is acceptable or whether waiting for successive solutions is warranted.

[1]IBM ILOG CPLEX Optimization Studio http://www-03.ibm.com/software/products/en/ibmilogcpleoptistud

[2]Gurobi Optimization, Inc. http://www.gurobi.com

4.3 Training Dataset

I demonstrate COVAS in the context of a real-world ASMD problem[3], described in Chapter 3. COVAS trains the apprenticeship scheduler using a dataset collected from military domain experts playing a serious game that emulates the ASMD problem as formulated in the previous section. I considered a specific level within the game that requires players to defend against a randomized enemy attack in which 10 missiles are fired at the player's ship from multiple directions, and the player has access to a limited quantity of five different types of soft-kill weapons to divert these missiles. Although the missile bearings and launch times are fixed, the seeking behavior of the missile is not known a priori.

I used the same ASMD dataset of 311 games played by 35 human players across 45 threat configurations, or "scenarios," from Chapter 3. I then sub-selected the single best demonstration from each of these 45 scenarios. The demonstrators included ASMD professionals with expertise ranging from "generally knowledgeable about the ASMD problem" to "domain experts" with professional experience or ASMD training.

I trained the apprenticeship scheduling algorithm using the following features employed in [89]: pointwise features for each action included the number of decoys of each type left for possible deployment (i.e., the ammunition). Pairwise features for each action included, for each decoy/missile pair (or null decoy deployment due to inaction), indicators for whether a decoy had been placed such that the missile was successfully distracted by that decoy, whether the missile would be lured into hitting the ship due to decoy placement, or whether the missile would be unaffected by decoy placement.

[3] I do not apply COVAS to the real-world healthcare problem because a mathematical model (i.e., an objective function and constraints) is not readily available and is an open area of research in healthcare. Ultimately, the goal in labor and delivery is to have mothers and babies leave the hospital to live long and healthy lives. Tying these downstream outcomes to upstream resource management decisions is less clear than the objective inherent in ASMD: survive the immediate attack. In future work, I propose exploring formulations of an objective function for labor and delivery that is supported by epidemiological evidence.

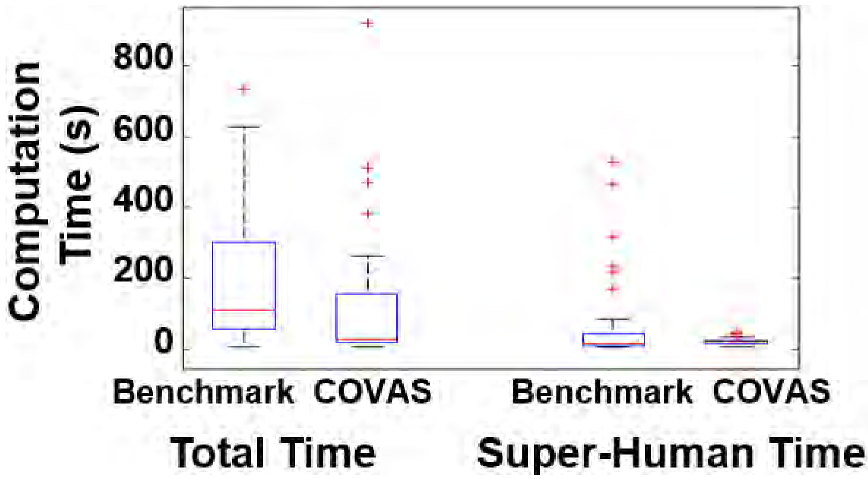

Figure 4-2: The total computation time for COVAS, as well as the time COVAS required to identify a solution superior to that resulting from a human expert's demonstration.

4.4 Results and Discussion

In this section, I empirically validate that COVAS is able to generate optimal solutions more efficiently than state-of-the-art optimization techniques. As a benchmark, I solve a pure MILP formulation (Equations 3.13 through 3.34) using Gurobi, which applies state-of-the-art techniques for heuristic upperbounds, cutting planes and linear-program (LP) relaxation lowerbounds. I set the optimality threshold, which defines when the optimization terminates, at 10^{-3}.

I note that, for the apprenticeship scheduling subroutine's schedulability test, I apply Equations 3.26 and 3.27 as a constraint satisfaction check when testing the feasibility of action τ_i^*, given by applying Equation 4.1. With regard to tasks within the apprenticeship scheduler's seed solution that are not satisfactorily completed, the MILP can leave those tasks incomplete to start by initially setting $V_m \leftarrow 0$.

4.4.1 Validation Against Expert Benchmark

I trained COVAS' apprenticeship scheduling algorithm on demonstrations of experts' solutions to unique ASMD scenarios (except for one "hold-out" scenario); I then tested COVAS on this hold-out scenario. I also applied a pure MILP benchmark on this scenario and compared the performance of COVAS to the benchmark. I generated one data point for each unique demonstrated scenario (i.e., leave-one-out cross validation) to validate the benefit of COVAS.

Figure 4-2 consists of two performance indicators: The total computation time required for the MILP benchmark and COVAS to solve for the optimal solution is depicted on the left; to the right is the computation time required for the benchmark and COVAS to identify a solution better than that given by a human expert. This figure indicates that COVAS is not only able to improve overall optimization time, but that it also substantially improves computation time for solutions that are superior to those produced by human experts. The average improvement in computation time with COVAS is 6.7x and 3.1x, respectively.

Next, I evaluate COVAS' ability to transfer prior learning to more challenging task sets. I trained on a level in the ASMD game in which a total of 10 missiles of varying types came from specific bearings at given times. I randomly generated a set of scenarios involving 15 and 20 missiles, with bearings and times randomly sampled with replication from the set of bearings used in the 10-missile scenario. Figure 4-3 depicts the computation time required by COVAS and the MILP benchmark to identify the optimal solution for scenarios involving 10, 15, and 20 missiles. I found that the average improvement to computation time with COVAS was 4.6x, 7.9x, and 9.5x, respectively. This evaluation demonstrates that COVAS is able to efficiently leverage the solutions of human domain experts to quickly solve problems twice as large as those the demonstrator provided for training.

4.4.2 Limitations and Future Work

COVAS is able to leverage expert scheduling demonstrations to speed up the computation of provable, globally optimal scheduling solutions. However, the approach is still limited by the quality of the demonstrations provided by the experts and the ability of the apprenticeship scheduling algorithm to generalize the information within those demonstrations. The MILP's computation time is expedited by tight lowerbound (i.e., an initial seed) provided by the apprenticeship scheduling algorithm.

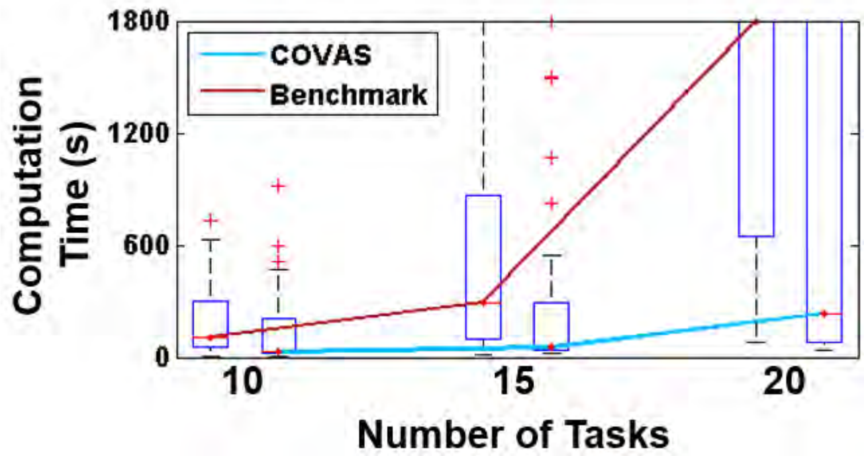

Figure 4-3: The total computation time needed for COVAS and the MILP benchmark to identify the optimal solution for the tested scenarios.

If the apprenticeship scheduling algorithm is unable to provide a tight lowerbound, the MILP's computation time may not be significantly improved. Future work will explore extensions to the apprenticeship scheduling algorithm to improve its ability to learn from noisy demonstrations. One approach could be to incorporate a trustworthiness metric á la [266] directly into the training of the classifier to uncover a latent action ranking. For example, instead of binary labels, I could reformulate the problem to be one of regression, where positive and negative labels are proportional and inversely proportional, respectively, to the fidelity of the demonstrator. Finally, despite the empirical benefit of COVAS, solving a MILP remains an exponentially-hard search problem.

4.5 Conclusions

In this work, I developed an approach to learning from human demonstrations to efficiently produce optimal solutions for complex real-world scheduling problems. I showed that policies learned from human experts can be used in conjunction with a MILP solver to substantially improve the efficiency of a branch-and-bound search for an optimal schedule. I validated my technique on a dataset collected from human experts solving an anti-ship missile defense problem, and showed that my approach can substantially improve upon solutions produced by human domain experts, at a rate up to 9.5 times faster than an optimization approach that does not incorporate human expert demonstration.

Chapter 5

Robot Embodiment as a Scheduling Apprentice[1]

5.1 Introduction

The goal of my thesis is to develop the computational techniques and human factors understanding to transform service robots from drones that must be explicitly tasked. Healthcare serves as a domain where these service robots are becoming the most widely utilized. They are deployed to improve operational efficiency by delivering and preparing supplies, materials, and medications [24, 63, 111, 178, 191]. While these systems exhibit autonomous capabilities for navigating from point to point, [175, 176], they do not operate with an understanding of patient status and needs, and must be explicitly tasked and scheduled. This can impose a substantial burden upon the nurse in charge of resource allocation, or the "resource nurse," – particularly within fast-paced hospital departments, such as the emergency or labor and delivery units. Proficient human labor nurses, on the other hand, are able to assist the resource nurse by anticipating her needs, act with some autonomy, and update the her as needed.

[1]This chapter is based upon work published at Robotics: Science and Systems. The citation is as follows:
Matthew C. Gombolay, Xi Jessie Yang, Brad Hayes, Nicole Seo, Zixi Liu, Samir Wadhwania, Tania Yu, Neel Shah, Toni Golen, and Julie A. Shah (2016, June). Robotic Assistance in Coordination of Patient Care. In Proc. Robotics: Science and Systems (RSS), Ann Arbor, Michigan, USA.

The need to explicitly task many service robots, or even novice labor nurses, may degrade the performance of a resource nurse [41, 56, 190], which has implications for both patient safety and the well-being of healthcare professionals [28, 129, 203, 229].

It is logical then, to give these system the ability to learn to adapt and anticipate the needs of their human counterparts via, e.g., apprenticeship scheduling. However, a robot that autonomously takes initiative when performing tasks may make poor decisions in the absence of oversight [146, 184, 237]. Decades of research in human factors cautions against fully autonomous decision making, as it contributes to poor human situational awareness and degradation in the human supervisor's performance [124, 196, 228, 258]. When integrating machines into human cognitive workflows, an intermediate level of autonomy is preferred [124, 258], in which the system provides suggestions to be accepted or modified by a human supervisor. Such a system would fall within the "4-6" range on the 10-point scale of Sheridan's levels of automation [196].

However, even when systems have limited autonomy, studies of human-automation interaction in aviation – another safety-critical domain – have shown that human supervisors can inappropriately trust in and rely upon recommendations made by automation systems [64]. For example, numerous aviation incidents have been attributed to human overreliance on imperfect automation [64]. Other studies have examined the effects of changes in system reliability, and found that it led to suboptimal control allocation strategies and reduced levels of trust in the systems [60, 61]. There is also evidence that suggestions provided by embodied agents engender overtrust and inappropriate reliance [215]. This concern is a critical barrier to fielding intelligent hospital service robots that take initiative to participate with nurses in decision making.

This chapter presents two novel contributions to the fields of robotics and healthcare. First, through human subject experimentation with physicians and registered nurses, this chapter presents the first known study involving experts working with an embodied robot on a real-world, complex decision making task comparing trust in and dependence on robotic versus computer-based decision support. Previous studies have focused on novice users and/or simple laboratory decision tasks [10, 58, 131, 154]. Our findings provide the first evidence that experts performing decision making tasks appear to be less susceptible to the negative effects of support embodiment, as trust assessments were similar in both the computer-based and robotic decision support conditions. Furthermore, embodiment yielded better performance on the part of the human counterpart as compared to a computer-based support specifically when the quality of recommendations would change. This work provides encouraging evidence that intelligent service robots can be safely integrated into the hospital setting.

Given evidence that robotic decision support has benefits for healthcare decision-making, we conduct a test demonstration in which a robot, using apprenticeship scheduling, assisted resource nurses on a labor and delivery floor in a tertiary care center. To assist in labor and delivery, the robot used computer vision techniques to read the current status of the labor floor, speech recognition to receive feedback from the resource nurse, and apprenticeship scheduling (Chapter 3) to generate scheduling recommendations. To my knowledge, this is the first investigation to field a robotic system in a hospital to aid in the coordination of resources required for patient care.

5.2 Experimental Investigation

In this section, I describe human-subject experimentation aimed at comparing trust in and dependence upon an embodied robot assistant versus computer-based decision support in a population of physicians and registered nurses. The participants interacted with a high-fidelity simulation of an obstetrics department at a tertiary care center. This simulation provided users the opportunity to assume the roles and responsibilities of a resource nurse, which included assigning labor nurses and scrub technicians to care for patients, as well as moving patients throughout various care facilities within the department. These care professionals work with the resource nurse, have an accurate model for her decision-making and can thus discern the appropriateness of the decision-support system's recommendations

The experiment employed a within-subjects design and manipulated two independent variables: *embodiment* – subjects received advice from either a robot or a computer, and *recommendation quality* – subjects received high- or low-quality advice. Each participant experienced four conditions, the quality of advice was blocked and the ordering of the conditions was counterbalanced in order to mitigate potential learning effects. Figure 5-1 depicts the experimental setup for the embodied condition.

5.2.1 Hypotheses and Measures

H1 *Rates of appropriate compliance with and reliance on robotic decision support will be comparable to or greater than those observed for computer-based decision support.* Objective measures of compliance and reliance were assessed based on the participants' "accept" or "reject" response to each decision support recommendation. Statistics on appropriate compliance, appropriate reliance, Type I (i.e., a miss) and Type II (i.e., a false alarm) errors were recorded.

H2 *Robotic decision support will be rated more favorably than computer-based decision support in terms of trust and other attitudinal measures.* Numerous studies have demonstrated that embodied and anthropomorphic systems are rated more favorably by users than computer-based interactive systems. I hypothesized that the robotic system in this study would elicit this favorable response (H2), while engendering appropriate rates of compliance and reliance (H1). This would indicate a positive signal for the successful adoption of a hospital service robot that participates in decision making. Subjective measures of trust and attitudinal response were collected via questionnaires administered to each participant after each of the four trials. Trust was assessed by a composite rating of seven-point Likert-scale responses for a commonly used, validated trust questionnaire [119]. Other attitudinal questions were drawn from [147] to evaluate personality recognition, social responses, and social presence in human-robot interaction, and were responded to on a 10-point Likert scale.

5.2.2 Materials and Setup

The experiments were conducted using a high-fidelity simulation of a labor and delivery floor. This simulation had previously been developed through a hospital quality improvement project as a training tool over a year-long, rigorous design and iteration process that included workshops with nurses, physicians, and medical students to ensure the tool accurately captured the role of a resource nurse. Parameters within the simulation (e.g. arrival of patients, timelines on progression through labor) were drawn from medical textbooks and papers and modified through alpha and beta testing to ensure that the simulation closely mirrored the patient population and nurse experience at the partner hospital.

An Aldebaran Nao was employed for the embodied condition (Figure 5-1). A video of the Nao offering advice to a participant with speech and co-speech gestures is viewable at `http://tiny.cc/NAORecommendation`. Participants received advice through synthesized speech under both the embodied and computer-based support conditions, using a male voice drawn from the Mary Text-to-Speech System (MaryTTS) [226]. The advice was also displayed as text in an in-simulation pop-up box under both conditions. The subject clicked a button in order to accept or reject the advice. These buttons were not clickable until the narration of the advice was complete; this narration took equal time in both conditions.

5.2.3 Experimental Procedure

Seventeen physicians and registered nurses participated in the experiment (one man and sixteen women). The participants were recruited from the partner hospital's obstetrics department via email and word-of-mouth.

First, participants provided consent for the experiment and watched an 8-minute tutorial video describing the labor and delivery floor simulation. The tutorial video is viewable at `http://tiny.cc/simTutorial`. Participants were instructed to play four simulated shifts on labor and delivery, with each iteration lasting 10 minutes, simulating a total of 4 hours on the labor floor. The computer or embodied sys-

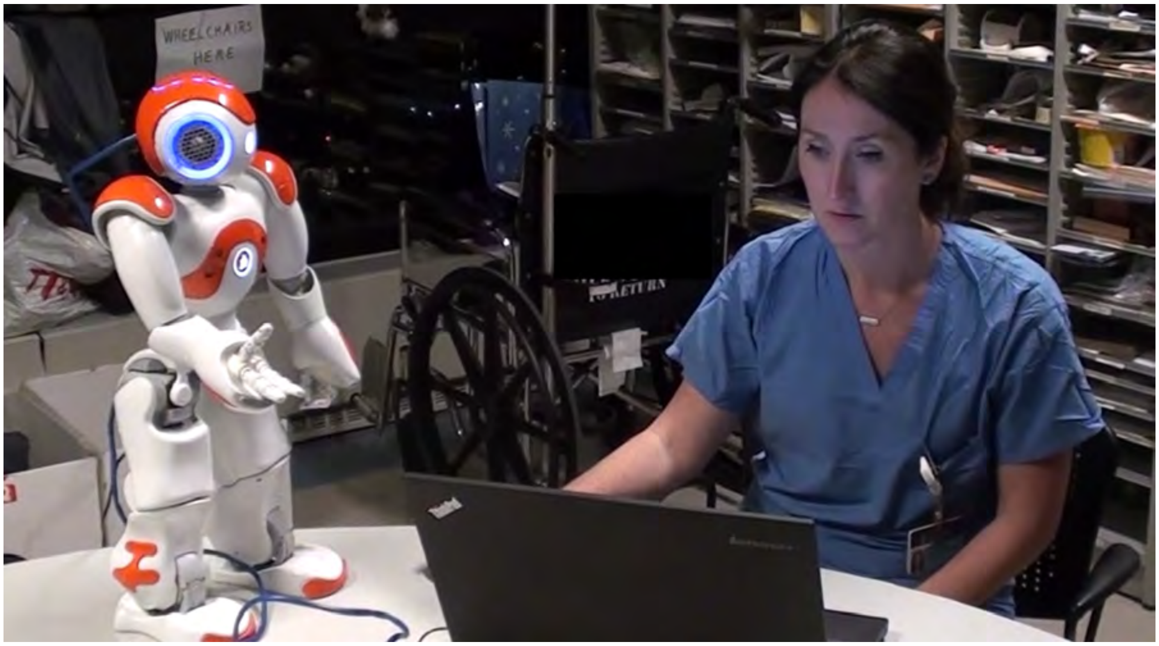

Figure 5-1: Experiment participant pictured receiving advice from the robotic decision support.

tem would interject during the simulation to make recommendations on which nurse should care for which patient, and on patient room assignments. Participants were asked to accept or reject the advice based on their own judgment. They were not informed whether the robotic or virtual decision support coach was providing high- or low-quality advice. Finally, after each of the four trials, participants were asked to rate their subjective experience via a set of Likert-scale questions, as described in Section 5.2.1.

5.3 Implementation of Decision Support

The challenge to decision support guidance is that the precise form of the objective function (Equation 3.35) that resource nurses optimize for is unknown. Prior work has indicated that domain experts are adept at describing the features (high-level, contextual and task-specific) used in their decision making, yet it is more difficult for experts to describe how they reason about these features [45, 206]. As such, I applied my apprenticeship scheduling technique from Chapter 3 to learn a set of heuristic scheduling policies from demonstrations of resource nurse decision making. I then applied these learned policies to produce advice for the computer-based and robotic decision support systems. Recall from Chapter 3 that apprenticeship scheduling was validated to generate advice that was accepted 90% of the time by the labor nurses and physicians.

To test the hypotheses, it was necessary to provide high- and low-quality advice. To generate high-quality advice, we merely apply the same apprenticeship scheduling technique from Chapter 3 vis-à-vis Equation 5.1 to find the action that has the highest cumulative probability of being more important than all other actions.

$$\tau_i^* = \underset{\tau_i}{\operatorname{argmax}} \sum_{\tau_x} f_{priority}(\tau_i, \tau_x) \tag{5.1}$$

To generate low-quality advice, I employed two separate methods: one to generate low-quality, and potentially *infeasible* advice, and one to generate low-quality but *feasible* advice. I make the distinction between infeasible and feasible advice for the following reason: In order to measure the reliance of the nurses on the generated advice, it would be necessary to have nonzero rates of false alarms and misses. If I merely used infeasible advice, the task of identifying the quality of that advice may be trivially simple. However, low-quality but feasible advice may require a more discerning eye. Thus, I presented participants with both types of low-quality advice.

The first method was applied to offer low-quality, potentially infeasible advice (e.g., assign a patient to an already-occupied room). This advice was generated by minimizing Equation 3.47, instead of maximizing it, as shown in Equation 5.2.

$$\tau_i^* = \underset{\tau_i}{\operatorname{argmin}} \sum_{\tau_x} f_{priority}(\tau_i, \tau_x) \tag{5.2}$$

The second method was applied to offer low-quality, feasible advice (e.g., assign a post-operating patient to triage). This advice was generated by evaluating Equation 5.2 after filtering the space of possible actions to include only feasible actions (per the constraints in Equations 3.36 through 3.44 from Chapter 3). We repeat these equations here for convenience.

$$M\left(2-{}^{t}A^{a}_{\tau_i^j}-H_{\tau_i}\right)\geq -U_{\tau_i^j}+{}^{t}G^{a}_{\tau_i^j}\geq M\left({}^{t}A^{a}_{\tau_i^j}+H_{\tau_i}-2\right),\forall\tau_i^j\in\boldsymbol{\tau},\forall t \tag{5.3}$$

$$\sum_{\tau_i^j\in\boldsymbol{\tau}}{}^{t}G^{a}_{\tau_i^j}\leq C_a,\forall a\in A,\forall t \tag{5.4}$$

$$\sum_{r\in R}{}^{t}R^{r}_{\tau_i^j}\geq 1-M\left(1-H_{\tau_i}\right),\forall\tau_i^j\in\boldsymbol{\tau},\forall t \tag{5.5}$$

$$\sum_{\tau_i^j\in\boldsymbol{\tau}}{}^{t}R^{r}_{\tau_i^j}\leq 1,\forall r\in R,\forall t \tag{5.6}$$

$$ub_{\tau_i^j}\geq f_{\tau_i^j}-s_{\tau_i^j}\geq lb_{\tau_i^j},\forall\tau_i^j\in\boldsymbol{\tau} \tag{5.7}$$

$$s_{\tau_x^y}-f_{\tau_i^j}\geq W_{\langle\tau_i,\tau_j\rangle},\forall\tau_i,\tau_j\in\boldsymbol{\tau}|,\forall W_{\langle\tau_i,\tau_j\rangle}\in\boldsymbol{TC} \tag{5.8}$$

$$f_{\tau_x^y}-s_{\tau_i^j}\leq D^{rel}_{\langle\tau_i,\tau_j\rangle},\forall\tau_i,\tau_j\in\boldsymbol{\tau}|\exists D^{rel}_{\langle\tau_i,\tau_j\rangle}\in\boldsymbol{TC} \tag{5.9}$$

$$f_{\tau_i^j}\leq D^{abs}_{\tau_i},\forall\tau_i\in\boldsymbol{\tau}|\exists D^{abs}_{\tau_i}\in\boldsymbol{TC} \tag{5.10}$$

Recommendations for the low-quality condition were produced by randomly selecting between these two methods in order to mitigate ordering effects. Within one simulated shift, the advice would either be all high-quality or all low-quality. We hypothesized the existence of a delay in recognizing the quality of advice and wanted each, separate shift to reach an equilibrium state of accepting and rejecting advice.

5.4 Results

This section reports the results of statistical testing of the experiment's hypotheses. Statistical significance is set at the $\alpha = 0.05$ level.

5.4.1 Analysis & Discussion of H1

Objective measures of compliance and reliance were assessed based on the participant's "accept" or "reject" responses to each decision support recommendation. Statistics on hits, misses, false alarms, and correct rejections are shown in Table 5.1. Results from a z-test for two proportions indicated no statistically significant

Table 5.1: Confusion matrix for participants shown as a raw count and percentage of the whole.

Robotic Decision Support		Response	
		Accept	Reject
Advice Quality	High	130 (44.5%)	16 (5.48%)
	Low	16 (5.48%)	130 (44.5%)

Virtual Decision Support		Response	
		Accept	Reject
Advice Quality	High	134 (45.3%)	14 (4.78%)
	Low	19 (6.48%)	126 (43.0%)

Table 5.2: Correct accept and reject decisions made with computer-based (C-Accept, C-Reject) versus Robotic (R-Accept, R-Reject) decision support, as a function of trial number, shown as a raw count and percentage of the whole.

	Trial Number			
	Bad Advice		Good Advice	
	1	2	3	4
C-Accept	5 (10.4%)	4 (6.7%)	41 (82.0%)	49 (92.5%)
R-Accept	9 (17.6%)	5 (9.6%)	43 (91.5%)	44 (93.6%)

	Trial Number			
	Good Advice		Bad Advice	
	1	2	3	4
C-Reject	2 (28.6%)	1 (2.8%)	11 (73.3%)	20 (87.0%)
R-Reject	3 (8.6%)	1 (10.0%)	21 (84.0%)	16 (94.1%)

difference in the Type II error rates for the robotic ($p_R = 13.1\%$) and computer-based ($p_C = 11.0\%$) decision support conditions ($z = 0.562, p = 0.713$), nor in the rates of correct "accept" responses to high-quality advice ($p_R = 90.5\%$, $p_C = 89.0\%$, $p = 0.713$) and "reject" responses to low-quality advice ($p_R = 86.9\%$, $p_C = 89.0\%$, $p = 0.287$) across the two conditions. Results from a TOST equivalence test using two z-tests for two proportions indicated that the rates of error, appropriate compliance, and appropriate reliance between the robotic and virtual decision support conditions were equivalent within 95% confidence.

The rates of Type I and Type II errors in the second and third trials, at the transition in advice quality (Table 5.2), were analyzed. Fisher's exact test found a significant difference in the rate of incorrect "accept" of low-quality advice (Type I error) across the second and third trials for the computer-based decision support (6.7% vs. 26.7%, $p = 0.046$), but not for the robotic support (9.6% vs. 16.0%, $p = 0.461$). A significant difference was also found in the rate of incorrect "reject" of high-quality advice (Type II error) across the second and third trials for the computer-based decision support (2.8% vs. 18.0%, $p = 0.040$), but not for robotic decision support (10.0% vs. 8.5%, $p \sim 1.0$). In other words, participants' rate of Type I errors associated with computer-based support increased significantly when participants had received high-quality advice in the previous trial. Similarly, the rate of Type II errors associated with computer-based support increased significantly when participants had received low-quality advice in the previous trial. No such significant differences were found for the robotic support conditions.

H1 Takeaway: These results support H1, in that Type I and Type II error rates were comparable between robotic and computer-based decision support. Furthermore, embodiment appeared to offer performance gains, as indicated by lower error rates after the quality of recommendation changed mid-experiment. These are encouraging findings because they provide evidence that a robotic assistant may be able to participate in decision making with nurses without eliciting inappropriate dependence. One potential rationale for these results is that experts may be less susceptible to the negative effects of embodiment, as has been documented for experienced users

Table 5.3: Subjective Measures Post-Trial Questionnaire with statistical significance. Questions 1-5 were responded to on a 7-point scale, and Questions 6-7 on a 10-point scale.

Trust and Embodiment in Human-Robot Interaction
1. I am suspicious of the system's intent, actions or outputs.
2. I think I could have a good time with this decision support coach.
3. People will find it interesting to use this decision support coach.
4. While you were interacting with this decision-support coach, how much did you feel as if it were an intelligent being?
5. While you were interacting with this decision-support coach, how much did you feel as if it were a social being?
6. Unsociable/Sociable.
7. Machine-Like/Life-Like.

interacting with anthropomorphic agents [195]. I note that this study was conducted with a stationary robot, in which movement was limited to co-speech gestures. Further investigation is warranted for situations in which experts interact with mobile service robots that participate in decision-making.

5.4.2 Analysis & Discussion of H2

A composite measure of trust was computed, as in [119]. Results from a repeated-measures ANOVA (RANOVA) demonstrated a statistically significant increase in the average rating for the decision support system under the high-quality advice condition ($M = 5.39$, $SD = 0.666$) as compared with the low-quality condition ($M = 3.49$, $SD = 1.26$) ($F(1, 14) = 46.3$, $p < 0.001$). However, a RANOVA yielded no statistically significant difference in trust between the robotic ($M = 4.41$, $SD = 1.32$) and computer-based ($M = 4.48$, $SD = 1.47$) embodiment conditions ($F(1, 14) = 0.450$, $p = 0.513$). Results from a TOST equivalence test, using two t-tests, indicated that subjects' trust ratings for the computer-based and robotic support were within one point of one another on a 7-point Likert Scale.

There were significant differences in the attitudinal assessment of the robotic versus computer-based decision support conditions for Questions 2, 3, 5, 6 in Table 5.3, indicating that participants rated the robotic system more favorably. The result was established using a two-way omnibus Friedman test, followed by pairwise Friedman tests. The test statistics for the pairwise Friedman tests were $p = 0.028, 0.007, 0.043$, and 0.005, respectively. Strikingly, there was not a single question (out of 37) for which participants rated the computer-based decision support significantly better than the robotic support.

The subjective perception of the character of the robot was significantly less sensitive to transitions in advice quality than the computer-based decision support. To reach this conclusion, the frequency with which the ratings of one embodiment condition subsumed the other, and vice versa, was computed. Specifically, $x_{R,L}$ is defined as the Likert-scale rating for a given question and a particular participant in the robotic low-quality advice condition, and likewise for the high-quality condition, $x_{R,H}$. The variables $x_{C,L}$, $x_{C,H}$ were similarly defined for the computer-based low- and high-quality conditions. The robotic condition was defined as subsuming the computer-based condition if either $\min(x_{R,L}, x_{R,H}) \leq \min(x_{C,L}, x_{C,H}) \leq \max(x_{C,L}, x_{C,H}) < \max(x_{R,L}, x_{R,H})$ or $\min(x_{R,L}, x_{R,H}) < \min(x_{C,L}, x_{C,H}) \leq \max(x_{C,L}, x_{C,H}) \leq \max(x_{R,L}, x_{R,H})$, and vice versa for the computer-based condition subsuming the robotic condition. A χ^2 test on the frequency of subsuming indicated that the participants' subjective evaluation according to Questions 1, 4, 6, 7 ($p = 0.045, 0.022, 0.005$ and 0.0043, respectively) changed more significantly under the computer-based condition than the robotic condition. There were no questions for which the response changed more significantly under the robotic condition versus the computer-based condition. In other words, the subjective assessment of the robot was more robust to advice-quality changes than the computer-based decision support. Further investigation is warranted to determine whether these effects persist over time as the users habituate to interaction with the robot.

H2 Takeaway: Our findings support H2 in that the robotic system was rated more favorably on attitudinal assessment than computer-based decision support, even as it engendered appropriate dependence. It is inevitable that a service robot will occasionally make poor-quality suggestions, and, advantageously, the robot engendered greater tolerance of errors than the computer-based decision support. These results indicate a positive signal for successful adoption of a robot that participates in a resource nurse's decision making.

5.4.3 Discussion on Embodiment

In the field of HRI, embodiment has become a ubiquitous topic of study; researchers have shown on numerous occasions that anthropomorphizing a computational algorithm affects human operators [105, 123, 135, 148, 149, 213, 240]. Yet, there is a lack of consensus for *why* embodiment has an effect. While I cannot speak for the effect of embodiment in all scenarios, I want to explore some ideas for why an embodied system (i.e., the NAO robot) improved rates of appropriate compliance with, reliance on, and engendered more positive attitudes toward the decision-support system as opposed to an unembodied system (i.e., computer-based). It is my goal that offering these ideas could inspire future, hypothesis-driven work to develop theories regarding embodiment's effect on human operators.

Conjecture 1: Information provided via an embodied system represents a more salient cue than information provided via an unembodied system.

In the experiment on the labor floor, a robotic decision-support system yielded more appropriate rates of compliance with and reliance on that decision-support system than a computer-based system. Specifically, participants' false alarm and miss rates increased significantly when the computer-based decision-support system changed its advice quality from high to lower; this effect was not significantly present for the robotic decision-support system.

I interpret this effect to mean that participants were more sensitive to – better able to detect – changes in the quality of the advice given by the robotic decision-support system. This objective measure of sensitivity showed a parallel to the subjective measures. Specifically, the magnitude of the change in people's attitudes towards the robot, as measured by a set of Likert items, was more significant than the change in their attitudes towards the computer-based decision-support system as a function of the advice quality of that system. In other words, people's opinion of the robot varied widely as a function of its advice quality changed, but not so for the computer-based system.

Anecdotally, I found the nurses on the labor floor to be inundated with computer-based information, including a deluge of false alarms and non-critical alerts. I believe, to a large extent, nurses have been forced to develop a level of indifference towards information delivered via computer screen. Instead, nurses tend to rely on human-generated information; nurses are alerted in-person by residents, physicians, and even their patients, when a need arises. Nurses know that, if a person has come to them with information, it must be important. Otherwise, that person would not have interrupted his or her current job to come speak with that nurse.

I believe that the robotic agent elicits more careful attention from the human operator because nurses have been tuned to respond to information from a physical agent (e.g., another nurse) as opposed to computer-based information, which, as noted above, is commonly unhelpful. To explore this idea further, one could conduct an experiment in which participants are primed by receiving unhelpful advice from a human, and helpful advice from a computer-based system (or vice versa). After the participants are primed, one then repeats a similar experimental protocol as the one conducted in this paper. I would hypothesize that participants primed by receiving bad advice from a human would be more sensitive to the advice from a computer-based system; similarly, participants primed with bad advice from a computer would be more sensitive to advice from a human.

Conjecture 2: People would rather work with an embodied system because a robot is more akin to a person than is a mere computer.

Nurses working with our robot were found to have more positive attitudes towards the NAO as compared to a computer-based decision-support system; they indicated they would prefer to work with the robot rather than the computer-based system. I believe this attitude to be true because a human would rather work with a teammate that is more similar to him- or herself. This belief is grounded in the notion of "homophily." Psychology, sociology, and human factors researchers have studied homophily, which is the idea that people prefer things (i.e., other people) that are similar to themselves [132, 127, 166, 170, 216, 261]. I believe people find it easier to understand and communicate with a robotic agent because it better approximates how people interact with other people than does a computer-based system.

One could conduct a relativity simple experiment which would compare degrees of anthropomorphization of the robotic agent, similar to the work of Pak et al. [195]. However, I would argue that changing a robots form factor to be more or less human-like, but still basically human-shaped, does not capture the significant difference between an animated robot and an inanimate computer.

A more grandiose proposal would be to explore the following hypothesis: People's attitudes towards an animal (i.e., a mouse, cat, human, etc.) is proportional to their attitudes towards a robot with the same form factor. One could conduct an experiment with robots of various animal form factors in which the robots assist in cognitive tasks. Then, measure the human participants' attitudes, and compare those attitudes to trials in which the participant interacts with the real animals. However, there are two potential limitations. The first limitation is technical – the ability to approximate the form factors of certain animals may be greater in some animals than in others; such discrepancies in the realism of the approximation could confound the experimental outcomes. Second, attitudes towards an agent assisting in a cognitive task (e.g., helping to make a scheduling decision) may not be directly comparable to attitudes towards that agent performing other tasks (e.g., a cat meowing in an attempt to be petted). This experiment proposal is merely a starting point.

There are likely many other ways to potentially explore this conjecture, and future researchers should not be limited by my experimental design as initially offered.

5.5 Pilot Demonstration of a Robotic Assistant on the Labor and Delivery Floor

Based on the positive results of this experiment, a pilot demonstration was conducted in which a robot assisted resource nurses on a labor and delivery floor at a tertiary care center. **Vision System:** In the experiments, the statuses of patients, nurses,

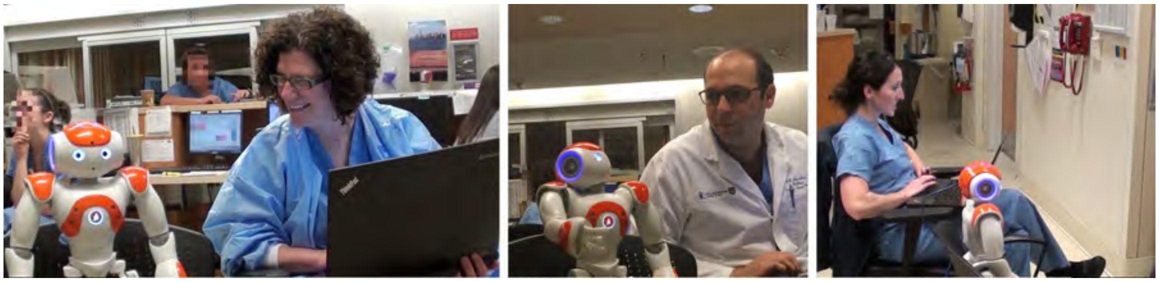

Figure 5-2: Images of the robot system in action on the labor floor.

and beds were provided and updated in the simulation. In contrast, nurses and robots on a real labor floor must read handwritten information off of a whiteboard (i.e., "dashboard") depicted in Figure 5-4. Extracting and parsing this information autonomously with high accuracy and reliability presents a substantial technical challenge. Two assumptions to address this challenge. 1) the set of physician and nurse names is closed and known in advance, and 2) patient names are transcribed for the robot upon patient arrival.

5.5.1 Robot System Architecture

As shown in Figure 5-3, the system was comprised of subsystems providing the vision, communication and decision support capabilities.

In this demonstration, the structured nature of the dashboard was leveraged to introduce priors that ensured patient information was interpretable. Rows on the dashboard indicate room assignments, while columns indicate patient parameters (e.g., attending physician, gestational age, etc.). Once the robot captured an image of the dashboard on the labor and delivery floor, Canny edge detection operator [35] and Hough transformation [71] were applied to isolate the handwriting in individual grid cells, as shown in Figure 5-5. The contents of each grid cell were processed using a classification technique appropriate to the data type therein. Numeric fields were parsed using a Convolutional Neural Network (CNN)[2] trained on MNIST data, while alphabetical fields with known sets of possible values (e.g., attending physician, nurse

[2]Thanks to Mikhail Sirontenko for developing this package, which is available at `https://sites.google.com/site/mihailsirotenko/projects/cuda-cnn`.

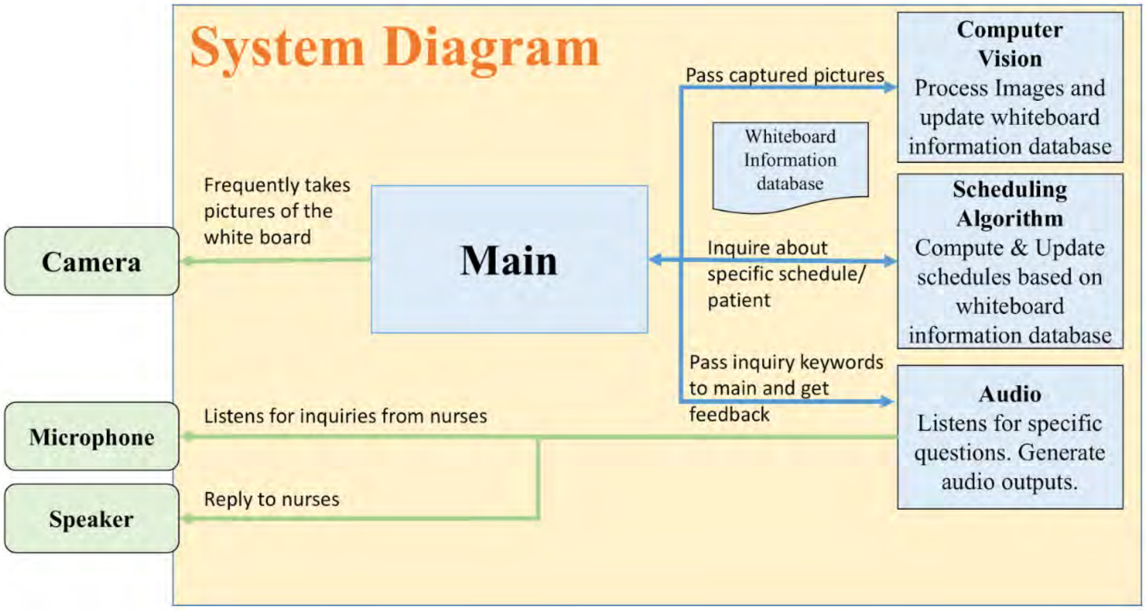

Figure 5-3: This figure depicts the service robot's system architecture.

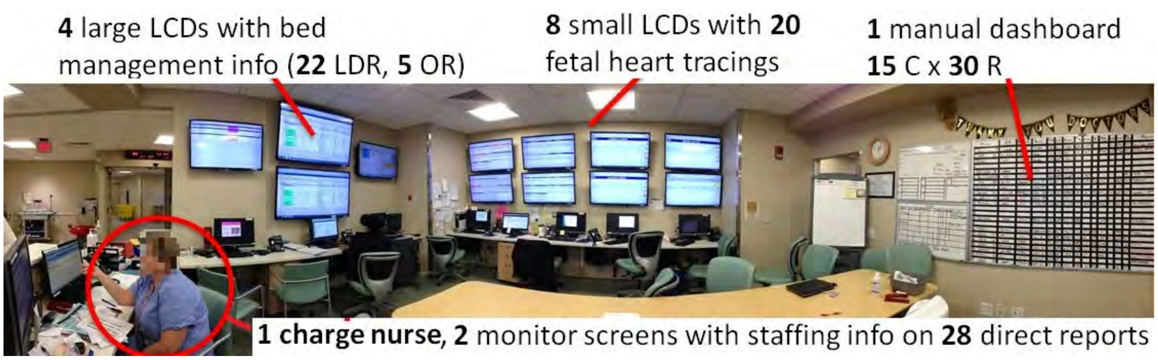

Figure 5-4: A resource nurse must assimilate a large variety and volume of information to effectively reason about resource management for patient care.

names) were parsed using a multi-class CNN trained on handwriting[3].

Handwriting samples (28 uniquely written alphabets) were used as a basis for generating classifier training data. Fonts were created from the provided samples and used (along with system fonts) to create a large set of binary images containing samples of nurse names. These synthetic writing samples were constructed with a range of applied translations, scalings, and kerning values within a 75x30 pixel area.

The vision system was used to determine the current status of patient-nurse allo-

[3]The CNN was constructed with the following architecture: 75x30 input layer → 5x5 kernel convolution layer → 2x2 kernel maxpool layer → 5x5 kernel convolution layer → 2x2 kernel maxpool layer → 100 node dense layer → classification layer.

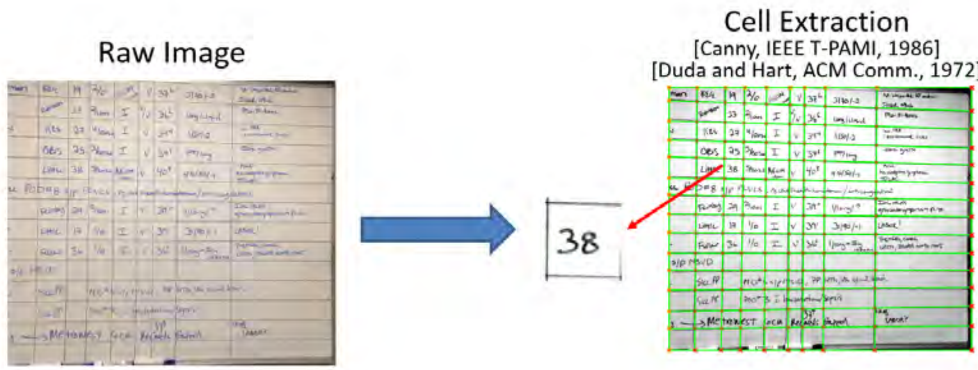

Figure 5-5: This figure depicts the cell extraction process.

cations, nurse role information, and room usage. Prior to deployment, a validation of the vision system was conducted; the recognition system was found to correctly classify handwritten samples across 15 classes (names) with a 97.8% accuracy. These results were obtained without performing any environmental manipulations (adjusting lighting, using high-resolution cameras, etc.). In the pilot deployment, the vision system assisted humans with transcription of patient data.

Communication: CMUSphinx [1] was employed for robot speech recognition. To achieve high performance in a live setting, a list of template-based phrases a user might utter, such as "Where should I move the patient in room [#]?" or "Who should nurse [*Name*] take care of?" were defined. All possible instantiations were enumerated based on information available a priori (e.g., the list of nurse names). Levenshtein distance [153] was computed to infer the phrase most likely uttered by the speaker, and the appropriate corresponding query was issued to the decision support system.

Decision Support: The live pilot demonstration of the robot used the same mechanism for generating decision support as that it used during the experiments. However, unlike the experiments, the decision support system's input was taken from the vision subsystem, and the user query from the communication subsystem. The set of possible actions to be recommended was filtered according to the query as recognized by the communication subsystem. For example, if the user asked, "Where should I move the patient in room 1A?," actions that would change nurse assignments were not considered. The recommended action was communicated to the user via text-to-speech software.

Feedback from Nurses and Physicians: A test demonstration was conducted on the labor floor (Figure 5-2). Three users interacted with the robot over the course of three hours. Ten queries were posed to the robot; seven resulted in successful exchanges and three failed due to background noise. A live recording of the demo can be seen at `http://tiny.cc/RobotDemo`. After interacting with the robotic support, User 1, a physician, said "I think the [robot] would allow for a more even dispersion of the workload amongst the nurses. In some hospitals ... more junior nurses were given the next patient ... more senior nurses were allowed to only have one patient as opposed to two." User 2, a resource nurse said, "New nurses may not understand the constraints and complexities of the role, and I think the robot could help give [the nurse] an algorithm ... that she can practice, repeat, and become familiar with so that it becomes second nature to her." User 3, a labor nurse offered, "I think you could use this robot as an educational tool."

5.6 Conclusion

This chapter addresses two barriers to fielding intelligent hospital service robots that take initiative to participate with nurses in decision making. Through experimental investigation, experts performing decision making tasks were found to be less susceptible to the negative effects of support embodiment. Further, based on the previous two findings, a first successful test demonstration was conducted in which a robot assisted resource nurses on a labor and delivery floor in a tertiary care center.

Chapter 6

Situational Awareness, Workload, and Workflow Preferences with Service Robots

6.1 Introduction

Thus far, I have developed and demonstrated a novel computational technique that enabled a hospital service robot to assist in the coordination of patient care. Chapter 5 investigated the effect of the embodiment of apprenticeship scheduling, which is an important variable affecting the human factors of human-robot teaming. While an embodied teammate was found to have positive effects on nurses and physicians in terms of trust and reliance, embodiment is but one variable. This chapter investigates three, quintessential facets of human factors: situational awareness, workload, and workflow preferences. Failing to consider these factors when introducing automation into a human environment has been shown to have serious consequences [74, 75, 77, 125, 214].

To begin, situational awareness has been defined as the ability to perceive, comprehend, and project the state of an environment [75]. Loss of situational awareness while operating highly autonomous systems has accounted for hundreds of deaths in commercial and general aviation (e.g., [179, 180, 181, 182, 183]). Humans must maintain their situational awareness in order to effectively take control of a job typically performed by an autonomous machine in the event that that machine fails. Second, workload assignment is another key issue in human factors [198, 233, 249, 257]. It has been shown in prior work that human performance is highly dependent upon workload [198, 204, 233, 249, 257]: A workload that is too heavy or too light can degrade performance and contribute to a loss of situational awareness [204, 249]. Third, understanding and incorporating workflow preferences is also essential for safe, effective, human-machine teaming [4, 104, 142, 144, 187]. In manufacturing, human teams can develop individualized workflow preferences that are not shared by other teams in the same environment; consequently, a member of one team may be unable to effectively replace a worker on another team without a period of adjustment.

This chapter reports the results from a series of three human subject experiments studying the factors of situational awareness, workload, and workflow preferences in the context of human-robot team coordination. First, this chapter investigates how situational awareness varies as a function of the degree of autonomy a robotic agent has during scheduling, and found that human participants' awareness of their team's actions decreased as the degree of robot autonomy increased. Given prior work indicating that humans typically prefer the robot to have greater autonomy [12, 87, 104, 113, 159], roboticists must balance the desire for increased automation and the performance improvements it yields with the risk for – and cost resulting from – reduced situational awareness. Second, this chapter studies how team fluency varies as a function of the workload (tasks not related to decision making about scheduling) given to a human team member by a robotic agent, and, third, the manner in which a robot should include the workflow preferences of its human teammates in the decision making process.

A roboticist or practitioner of multi-agent coordination might take the most straightforward approach by including the preferences of each human team member and balancing work assignments according to a given fairness metric. However, these experiments show that when the goal of including human team members' preferences is orthogonal to the goal of assigning each agent tasks in the way that most benefits the team's overall performance, people are usually amenable to relinquishing their preferred assignments for the sake of improved team fluency. The results of this experiment show that there is a relationship between humans' preferences, their utilization during task execution, and their perception of team efficiency. Participants felt more strongly that their teams performed the assigned tasks using the least possible amount of time, even though the schedule duration (makespan) was constant across all trials within participants.

6.2 Aims of the Experiment

Prior literature [44, 68, 81, 87, 101] has shown the potential advantages of providing a robotic teammate with greater autonomy, and recent work in the realm of shared-autonomy in scheduling has extended these findings. Such works have indicated that a robot generates a schedule more quickly and a team is able to complete assigned tasks more efficiently when the schedule is generated by the robotic agent alone as opposed to when a human team member assists in the scheduling process [91]. Furthermore, participants in prior experiments have readily stated they would prefer working with a robotic teammate with a greater degree of autonomy [87].

However, this recent work provides an incomplete picture. For example, the ramifications of conceding autonomy to a robotic agent, especially in environments where human team members might have to reallocate work manually due to an environmental disturbance that the robot is unable to consider, is unclear. Also, it is unknown whether the *way* in which a robot schedules a team (e.g., whether the robot happens to assign tasks to participants who prefer them) substantially affects the participants' experiences. Finally, prior work has not shown whether the amount of work assigned by the robot in such a human-robot teaming scenario results in a suitable workload for human teammates.

This chapter presents three experiments to better understand 1) whether situational awareness degrades when the robotic agent has a greater degree of control over scheduling decisions, 2) how a robotic agent should schedule tasks for a human-robot team given the humans' workflow preferences, and 3) whether there is a trade-off between the degree to which human team members' scheduling preferences are included in the scheduling process and the effective utilization of those workers.

6.2.1 Experiment: Situational Awareness in Mixed-Initiative Human-Robot Teaming

Prior work in human factors has indicated that there are significant consequences associated with ceding decision making initiative to an autonomous agent [74, 75, 125, 126, 214]. Chiefly, the human counterpart can experience a decline in situational awareness. This phenomenon has been observed in a variety of domains, including telerobotics [126]. Thus, an experiment was proposed to serve as the first such investigation in the setting of human-robot teaming using mixed-initiative[1] scheduling, with the human and robot sharing scheduling responsibilities.

Independent Variable

To determine the potential consequences of providing a robotic teammate with greater autonomy over scheduling decisions, a novel human subject experiment was conducted, which consisted of three team members: a robot, a human subject, and a human assistant (i.e., a confederate) who were required to complete a series of fetching and building tasks. In this experiment, the independent variable was the allocation of authority over scheduling decisions; this independent variable had three levels, or conditions:

- *Manual control*: The human subject decides who will perform each of the tasks.

[1] I use the term "mixed-initiative" to describe that, in some of the experimental conditions presented in this chapter, the human is responsible for some or all of the high-level task allocation decisions, while the robot is responsible for the remaining task allocation decisions as well as all of the low-level sequencing decisions. However, the term mixed-initiative can be used in different contexts [80].

- *Semi-autonomous control*: The human subject decides which tasks he or she will perform, and the robot assigns the remaining tasks to itself and the human assistant.

- *Autonomous control*: The robot decides who will perform each of the tasks.

Hypothesis

The following hypothesis was established:

Hypothesis 1: Participants' situational awareness will be poorer when the robotic teammate has greater autonomy over scheduling decisions.

Dependent Variables

An experiment using the Situation Awareness Global Assessment Technique, or SAGAT [74] was conducted to test **Hypothesis 1**. SAGAT was designed to measure the situational awareness of a pilot in an aircraft cockpit. During an experiment in which a pilot operated a simulated aircraft, the experimenter blanked out the information displays and the pilot was required to recall vital information about the state of the aircraft.

This protocol has disadvantages. For example, halting the experiment to question the subject is highly intrusive and could lead to a decline in performance when the subject must resume flying the aircraft. Also, the responses are highly dependent upon the subject's ability to remember information, which decays as a function of time – over the course of a long test, the subject may begin to forget important pieces of information about the system's state. In the experimental design, the same test was applied, in the same manner, to all participants; therefore, any such negative effect would be balanced across experimental conditions. Furthermore, the SAGAT test was not repeated. The test was only administered once during the experiment, which concluded after this administration.

For the SAGAT test, a set of objective and subjective measures, as shown in Table 6.1, was employed. The objective measures evaluated the accuracy of the participants'

Table 6.1: This table depicts the post-trial questionnaire administered to participants for the experiment measuring situational awareness as a function of the level of autonomy over scheduling decisions given to the robotic teammate. Participants responded to Questions 1, 5, 9, 13, and 17 using the response form shown in Table 6.2. Participants responded to Questions 2-4, 6-8, 10-12, 14-16, and 18-20 using a Likert response format consisting of "Strongly Disagree," "Weakly Disagree," "Neutral," "Weakly Agree" and "Strongly Agree."

Current Actions
1. What is each team member currently doing? *(Circle nothing if the team member is idle).*
2. I am aware of what the robot co-leader is doing.
3. I am aware of what the human assistant is doing.
4. I am aware of what I am doing.
Preceding Action
5. Which task did each team member last complete prior to the current task? *(Circle nothing if the team member has not yet completed a task.).*
6. I am aware of which task the robot co-leader just did.
7. I am aware of which task the human assistant just did.
8. I am aware of what I just did.
Past Schedule
9. Please list tasks each team member has completed. List the tasks in the order in which they were completed by writing 1 for the first task, 2 for the second task, and so forth.
10. I am aware of which tasks the human/robot co-leader has completed.
11. I am aware of which tasks the human assistant has completed.
12. I am aware of which tasks I have completed.
Future Schedule
13. Which tasks will each team member complete in the future? (Circle one task in each row to show which team member will complete which task in the future.)
14. I am aware of which tasks the human/robot co-leader will do in the future.
15. I am aware of which tasks the human assistant will do in the future.
16. I am aware of which tasks I will do in the future.
Dynamic Re-Scheduling
17. Given the work that has already been completed, who do you anticipate will complete the remaining tasks if the human/robot co-leader was no longer available?
18. I am aware of the team's schedule.
19. If I had to come up with a new schedule for the team, I would know enough.
20. If I had to come up with a new schedule for the team, I would do a good job.

Team Leader (You)	**Human Assistant**	**Robot Co-Leader**
Fetch B	Fetch B	Fetch B
Fetch C1	Fetch C1	Fetch C1
Fetch C2	Fetch C2	Fetch C2
Build A	Build A	Build A
Build B	Build B	Build B
Build C1	Build C1	Build C1
Build C2	Build C2	Build C2
Fetch E	Fetch E	Fetch E
Fetch F	Fetch F	Fetch F
Fetch G	Fetch G	Fetch G
Build D	Build D	Build D
Build E	Build E	Build E
Build F	Build F	Build F
Build G	Build G	Build G

Table 6.2: This table depicts the response format for the post-test questionnaire shown in Table 6.1.

perceptions of the state of the human-robot team; the subjective measures were paired with the objective measures to evaluate the participants' confidence in their answers to the objective questions.

Participants responded to the objective questions (i.e., Questions 1, 5, 9, 13, and 17) using the template-based response format shown in Table 6.2, and responded to subjective questions (i.e., questions 2-4, 6-8, 10-12, 14-16, and 18-20) according to a 5-point Likert response format consisting of "strongly disagree," "weakly disagree," "neutral," "weakly agree," and "strongly agree." The later questions were applied to gain insight into the participants' subjective perception of their situational awareness.

Table 6.2 depicts each individual subtask that could be assigned to each team member. (I describe the nature of these subtasks in the subsequent description of the experiment design.) However, I note for clarity that the task set consisted of fetching and building tasks A, B, C1, C2, D, E, F, and G, such that the fetch and build subtasks for C1 were required to be completed before the fetch and build subtasks for C2 could begin. The table does not include fetch operations for A and D because the experiment began with kits A and D already fetched. This condition increased the number of possible actions the human agents could take at the start of the experiment.

To test Hypothesis 1 via objective measures, a metric, called the "SA Score," was defined to assess how well participants are able to provide the desired information for each question in Table 6.1. The SA Score for each team member was computed according to Equation 6.1, and the overall SA Score was computed for the whole team according to Equation 6.2. In these equations $S^a_{response}$ is the set of tasks the subject reported for agent a for a given question, and $S^a_{correct}$ is the correct set of tasks for agent a for that same question. In this manner, sets $S^a_{response}$ and $S^a_{correct}$ were obtained for each agent and for each of the objective Questions 1, 5, 9, 13, and 17.

$$\text{SA Score for Agent } a := |S^a_{response} \setminus S^a_{correct}| + |S^a_{correct} \setminus S^a_{response}| \tag{6.1}$$

$$\text{SA Score for Team} := \sum_{a=1}^{n} (\text{SA Score for Agent } a) \tag{6.2}$$

In essence, Equation 6.1 counts the number of mistakes, false positives (incorrect tasks identified in the response) and false negatives (correct tasks not identified in the response). The team's SA score is a sum of the individual SA scores. This work assumed that the subject's situational awareness of each team member is equally important. A perfect score is equal to zero, and the worst possible score is equal to the total number of fetch and build tasks (14).

Let us consider an example in which the correct answers are as follows: Subject - $S^{participant}_{correct} = \{\text{Fetch B, Build C1}\}$, Human Assistant - $S^{asst.}_{correct} = \{\text{Fetch C1, Build A}\}$, and Robotic Agent - $S^{robot}_{correct} = \{\emptyset\}$. Let us say the subject provided the following answer: Subject - $S^{subject}_{response} = \{\text{Fetch B, Build A}\}$, Human Assistant - $S^{asst.}_{correct} = \{\text{Fetch C1, Build D, Build G}\}$, and Robotic Agent - $S^{robot}_{correct} = \{\text{Fetch E}\}$. The SA score would then be calculated as follows:

$$
\begin{aligned}
\text{SA Score for subject} &= |S^{participant}_{response} \backslash S^{participant}_{correct}| + |S^{participant}_{correct} \backslash S^{participant}_{response}| \\
&= |\{\text{Build A}\}| + |\{\text{Build C1}\}| \\
&= 2 \\
\text{SA Score for asst.} &= |S^{asst.}_{response} \backslash S^{asst.}_{correct}| + |S^{asst.}_{correct} \backslash S^{asst.}_{response}| \\
&= |\{\text{Build D, Build G}\}| + \{\text{Build A}\}| \\
&= 3 \\
\text{SA Score for robot} &= |S^{robot}_{response} \backslash S^{robot}_{correct}| + |S^{robot}_{correct} \backslash S^{robot}_{response}| \\
&= |\{\text{Build A}\}| + |\{\emptyset\}| \\
&= 1 \\
\text{SA Score for Team } &= \sum_{a=1}^{n} (\text{SA Score for Agent } a) \\
&= 2 + 3 + 1 \\
&= 6
\end{aligned}
$$

6.2.2 Experiment: Workflow Preferences

An experiment was proposed to understand how the robot's inclusion of human team members' preferences (e.g., a subject preferring to complete build tasks over fetch tasks) for completing particular tasks affects the relationship between the human and robotic agents.

Independent Variable

The independent variable was the degree to which participants' preferences were respected by the robotic teammate when scheduling. Three experimental conditions for this variable using a within-participants experiment design:

- *Positive*: The robot generates a schedule incorporating the preferences of the subject.
- *Neutral*: The robot ignores the preferences of the subject.
- *Negative*: The robot schedules according to the opposite of the preferences stated by the subject.

Hypothesis

The following hypothesis was established:

Hypothesis 2: Participants would prefer to work with a robotic teammate that incorporates their scheduling preferences than with one that is unaware of their preferences, and participants would prefer to work with a robotic teammate that is ignorant to their preferences than with one that actively schedules against their preferences.

Dependent Variables

To test this hypothesis, a within-participants experiment was conducted in which all participants experienced each of the three conditions once, and received a post-trial questionnaire after experiencing each condition. This questionnaire consisted of 21 Likert statements, as shown in Table 6.3. Hoffman previously developed and validated the questions drawn from the "Robot Teammate Traits" and "Working Alliance for Human-Robot Teams" surveys [103]. The later survey is a derivative of the "Working Alliance Inventory," originally developed and validated by Horvath et al. [110].

Participants also responded to a questionnaire upon completing the tasks under each condition, as shown in Table 6.4. This questionnaire gathered demographic information and included three additional Likert statements summarizing the experience of the participants, along with two open-ended questions.

This questionnaire is not balanced. The number of positive prompts (e.g., "I believe the robot likes me.") outweighed the number of negative prompts (e.g., "I feel uncomfortable with the robot."). However, potential bias arising from an unbalanced survey is mitigated since the same questionnaire is administered in each condition.

Table 6.3: Subjective Measures – Post-Trial Questionnaire

Robot Teammate Traits
1. The robot was intelligent.
2. The robot was trustworthy.
3. The robot was committed to the task.
Working Alliance for Human-Robot Teams
4. I feel uncomfortable with the robot. (reverse scale)
5. The robot and I understand each other.
6. I believe the robot likes me.
7. The robot and I respect each other.
8. I feel that the robot worker appreciates me.
9. The robot worker and I trust each other.
10. The robot worker perceives accurately what my goals are.
11. The robot worker does not understand what I am trying to accomplish. (reverse scale)
12. The robot worker and I are working toward mutually agreed-upon goals.
13. I find what I am doing with the robot worker confusing. (reverse scale)
Additional Measures of Team Fluency
14. I was satisfied by the team's performance.
15. I would work with the robot the next time the tasks were to be completed.
16. The robot increased the productivity of the team.
17. The team collaborated well together.
18. The team performed the tasks in the least time possible.
19. The robot worker was necessary to the successful completion of the tasks.
20. The human worker was necessary to the successful completion of the tasks.
21. I was necessary to the successful completion of the tasks.

6.2.3 Experiment: Workload vis á vis Workflow Preferences

The exclusive focus of the previous experiment was on modulating the degree to which scheduling preferences were included, and did not control for workload – rather, the experiment controlled for overall team efficiency (makespan). This experiment examined how team fluency varies as a function of the size of the workload assigned to a human by a robotic teammate while controlling for workflow preferences. The results section discusses how including participants' preferences in the scheduling process can decrease their workload, and, in turn, lead to decreased team fluency.

To isolate the effects of variation in a subject's workload, the inclusion of scheduling preferences and increasing the subject's workload were separated into two inde-

Table 6.4: Subjective Measures – Post-Test Questionnaire

Overall Preference
22. If the robot scheduled me to do the tasks I preferred, I would want to work with the robot again.
23. If the robot did not know my preferences when scheduling, I would want to work with the robot again.
24. If the robot scheduled me to do different tasks than what I preferred, I would want to work with the robot again.
Open-Response Questions
25. Which of the three scenarios did you prefer, and why?
26. If you were going to add a robotic assistant to a manufacturing team, to whom would you give the job of rescheduling the work, and why?

pendent variables. The hypothesis posits that decoupling workload from preferences results in a clearer understanding of the effects of varying workload – and, in turn, the inclusion of workflow preferences.

Independent Variables

This experiment considered two independent variables: 1) the participants' utilization, defined as the total amount of time the subject was occupied during execution of a particular schedule, and 2) the degree to which the robot respected participants' preferences during scheduling. Subject are considered as having high utilization if the majority of their time was spent working as opposed to being idle, and vice versa for low utilization. This experiment was comprised of a 2x2 within-participants design with the following four conditions, as shown in Table 6.5.

- *High Preference - High Utilization*: The robot generates a schedule incorporating the preferences of the participant and highly utilizes the participant.
- *High Preference - Low Utilization*: The robot generates a schedule incorporating the preferences of the participant and minimally utilizes the participant.
- *Low Preference - High Utilization*: The robot generates a schedule according to the opposite of the preferences of the participant and highly utilizes the participant.

2x2 Design	High Utilization	Low Utilization
High Preference	High Preference - High Utilization	High Preference - Low Utilization
Low Preference	Low Preference - High Utilization	Low Preference - Low Utilization

Table 6.5: This table depicts the four experimental conditions varying the two independent variables (the degree to which the scheduling preferences are included and the participants' utilization), each of which have two levels: high and low.

- *Low Preference - Low Utilization*: The robot generates a schedule according to the opposite of the preferences of the participant and minimally utilizes the participant.

Hypotheses

The following hypotheses were established for this experiment:

Hypothesis 3A: A participant's subjective assessment of their robotic teammate is favorably influenced by working with a robot that makes allocation decisions that incorporate their scheduling preferences, as opposed to decisions that contradict their preferences. (In contrast to H2, this hypothesis was assessed while controlling for the workload utilization of the participant.)

Hypothesis 3B: A participant's subjective assessment of their robotic teammate is favorably influenced by working with a robot that makes work allocation decisions that result in high utilization of the participant's time, as opposed to low utilization.

Dependent Variables

A within-participants experiment in which each participant experienced each condition once was conducted to test these hypotheses. As in the previous experiment, a post-trial questionnaire was administered after each of the conditions, as well as a post-test questionnaire after each participant completed all conditions. The investigating agent workload included four conditions; as such, participants responded to a total of four post-trial questionnaires. The same design for the post-trial (Table 6.3) and post-test questionnaires (Table 6.4) were used from the previous experiment.

6.3 Formal Problem Definition

The problem of scheduling a team of heterogeneous agents to complete a set of tasks with upper- and lowerbound temporal constraints and shared resources (e.g., spatial locations) falls within the XD [ST-SR-TA] class of scheduling problems, according to the comprehensive taxonomy defined by [139]. This class is one of the most computationally challenging in the field of scheduling. The XD [ST-SR-TA] class of problems is composed of tasks requiring one robot or agent at a time (single-robot tasks [ST]), robots/agents that perform one task at a time (single-task robots [SR]) and a time-extended schedule of tasks that must be built for each robot/agent (time-extended allocation [TA]). This time-extended schedule includes cross-schedule dependencies (XD) amongst the individual schedules of the agents; such dependencies arise, for example, when agents must share limited-access resources (e.g., physical locations).

I formulated an instance of this problem in order to develop an experiment task as a mixed-integer linear program, as depicted in Equations 6.3 through 6.13. This formulation serves as a common basis to model each of the three experiments. I subsequently discuss experiment-specific extensions.

$$\min z, z = g\Big(\{A^a_{\tau_i^j} | \tau_i^j \in \boldsymbol{\tau}, a \in A\},$$
$$\{J_{\langle \tau_i^j, \tau_x^y \rangle} | \tau_i^j, \tau_x^y \in \boldsymbol{\tau}\}, \{s_{\tau_i^j}, f_{\tau_i^j} | \tau_i^j \in \boldsymbol{\tau}\}\Big) \tag{6.3}$$

subject to

$$\sum_{a \in A} A^a_{\tau_i^j} = 1, \forall \tau_i^j \in \boldsymbol{\tau} \tag{6.4}$$

$$ub_{\tau_i^j} \geq f_{\tau_i^j} - s_{\tau_i^j} \geq lb_{\tau_i^j}, \forall \tau_i^j \in \boldsymbol{\tau} \tag{6.5}$$

$$f_{\tau_i^j} - s_{\tau_i^j} \geq lb^a_{\tau_i^j} - M\left(1 - A^a_{\tau_i^j}\right), \forall \tau_i^j \in \boldsymbol{\tau}, a \in A \tag{6.6}$$

$$s_{\tau_x^y} - f_{\tau_i^j} \geq W_{\langle \tau_i^j, \tau_x^y \rangle}, \forall \tau_i^j, \tau_x^y \in \boldsymbol{\tau} |, \forall W_{\langle \tau_i^j, \tau_x^y \rangle} \in \boldsymbol{TC} \tag{6.7}$$

$$f_{\tau_x^y} - s_{\tau_i^j} \leq D^{rel}_{\langle \tau_i, \tau_j \rangle}, \forall \tau_i^j, \tau_x^y \in \boldsymbol{\tau} | \exists D^{rel}_{\langle \tau_i^j, \tau_x^y \rangle} \in \boldsymbol{TC} \tag{6.8}$$

$$f_{\tau_i^j} \leq D^{abs}_{\tau_i^j}, \forall \tau_i \in \boldsymbol{\tau} | \exists D^{abs}_{\tau_i^j} \in \boldsymbol{TC} \tag{6.9}$$

$$s_{\tau_x^y} - f_{\tau_i^j} \geq M\left(A^a_{\tau_i^j} + A^a_{\tau_x^y} - 2\right)$$
$$+ M\left(J_{\langle \tau_i^j, \tau_x^y \rangle} - 1\right), \forall \tau_i^j, \tau_x^y \in \boldsymbol{\tau}, \forall a \in A \tag{6.10}$$

$$s_{\tau_i^j} - f_{\tau_x^y} \geq M\left(A^a_{\tau_i^j} + A^a_{\tau_x^y} - 2\right)$$
$$- M\left(J_{\langle \tau_i^j, \tau_x^y \rangle}\right), \forall \tau_i^j, \tau_x^y \in \boldsymbol{\tau}, \forall a \in A \tag{6.11}$$

$$s_{\tau_x^y} - f_{\tau_i^j} \geq M\left(J_{\langle \tau_i^j, \tau_x^y \rangle} - 1\right),$$
$$\forall \tau_i^j, \tau_x^y \in \boldsymbol{\tau} | R_{\tau_i^j} = R_{\tau_x^y} \tag{6.12}$$

$$s_{\tau_i^j} - f_{\tau_x^y} \geq -M\left(J_{\langle \tau_i^j, \tau_x^y \rangle}\right) \forall \tau_i, \tau_j \in \boldsymbol{\tau} | R_{\tau_i^j} = R_{\tau_x^y} \tag{6.13}$$

In this formulation, $A^a_{\tau_i^j} \in \{0, 1\}$ is a binary decision variable for the assignment of agent a to subtask τ_i^j (i.e., the j^{th} subtask of the i^{th} task); $A^a_{\tau_i^j}$ equals 1 when agent a is assigned to subtask τ_i^j and 0 otherwise. $J_{\langle \tau_i^j, \tau_x^y \rangle} \in \{0, 1\}$ is a binary decision variable specifying whether τ_i^j comes before or after τ_x^y, and $s_{\tau_i^j}, f_{\tau_i^j} \in [0, \infty)$ are the start and finish times of τ_i^j, respectively. $\boldsymbol{TC}$ is the set of simple temporal constraints relating task events. M is a large, positive constant used encode conditional statements as linear constraints.

Equation 6.3 is a general objective that is a function of the decision variables $\{A^a_{\tau_i^j} | \tau_i^j \in \boldsymbol{\tau}, a \in A\}$, $\{J_{\langle \tau_i^j, \tau_x^y \rangle} | \tau_i^j, \tau_x^y \in \boldsymbol{\tau}\}$ and $\{s_{\tau_i^j}, f_{\tau_i^j} | \tau_i^j \in \boldsymbol{\tau}\}$. Equation 6.4 ensures that each τ_i^j is assigned to a single agent. Equation 6.5 ensures that the duration of each $\tau_i^j \in \boldsymbol{\tau}$ does not exceed its upper- and lowerbound durations. Equation 6.6 requires that the duration of τ_i^j, $f_{\tau_i^j} - s_{\tau_i^j}$ is no less than the time required for agent a to complete τ_i^j. Equation 6.7 requires that τ_x^y occurs at least $W_{\langle \tau_i^j, \tau_x^y \rangle}$ units of time after τ_i^j (i.e., $W_{\langle \tau_i^j, \tau_x^y \rangle}$ is a lowerbound on the amount of time between the start of τ_x^y and the finish of τ_i^j).

Equation 6.8 requires that the duration between the start of τ_i^j and the finish of τ_x^y is less than $D^{rel}_{\langle \tau_i^j, \tau_x^y \rangle}$ (i.e., $D^{rel}_{\langle \tau_i^j, \tau_x^y \rangle}$ is an upperbound on the finish time of τ_x^y *relative* to the start of τ_i^j). Equation 6.9 requires that τ_i^j finishes before $D^{abs}_{\tau_i^j}$ units of time have expired since the start of the schedule (i.e., $D^{abs}_{\tau_i^j}$ is an upperbound on the latest *absolute* time τ_i^j can be finished). Equations 6.10 and 6.11 enforce that agents can only execute one subtask at a time. Equations 6.12 and 6.13 enforce that each resource R_i can only be accessed by one agent at a time.

The worst-case time complexity of a complete solution technique for this problem is dominated by the binary decision variables for allocating tasks to agents $(A^a_{\tau_i^j})$ and sequencing $(J_{\langle \tau_i^j, \tau_x^y \rangle})$, and the complexity is given by $O\left(2^{|A||\boldsymbol{\tau}|^3}\right)$, where $|A|$ is the number of agents and $|\boldsymbol{\tau}|$ is the number of tasks. Agent allocation contributes $O\left(2^{|A||\boldsymbol{\tau}|}\right)$, and sequencing contributes $O\left(2^{|\boldsymbol{\tau}|^2}\right)$.

6.4 Scheduling Mechanism

For all three experiments, I adapted a dynamic scheduling algorithm, called Tercio [91], to schedule the human-robot teams. Tercio is an empirically fast, high-performance dynamic scheduling algorithm designed for coordinating human-robot teams with upper- and lowerbound temporospatial constraints. The algorithm is designed to operate on a simple temporal network [177] with set-bounded uncertainty.

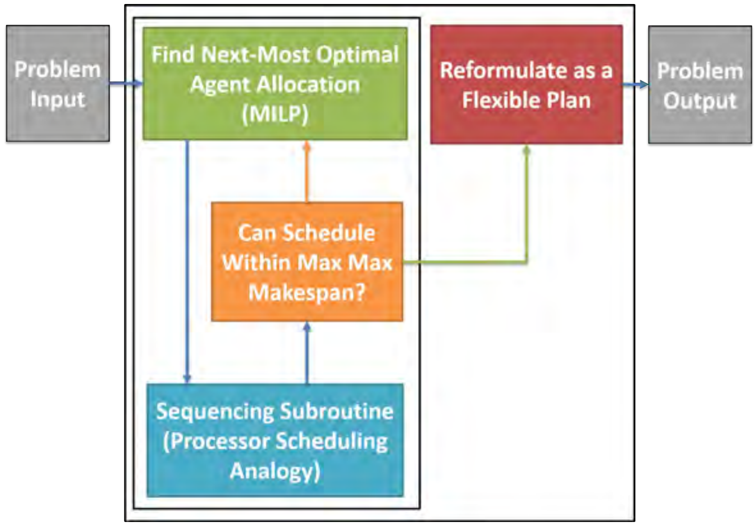

Figure 6-1: Tercio takes as input a temporal constraint problem and finds a satisficing, flexible schedule by utilizing an analytical schedulability test to ensure a feasible solution.

If the schedule's execution exceeds its set bounds, Tercio re-schedules the team [91].

I note that, for this experimental investigation, I did not employ apprenticeship scheduling as the basis for the robotic teammate's ability to make scheduling decisions. The premise of apprenticeship scheduling is to learn from human expert demonstration how to solve complex scheduling problems. However, for this experiment, I need to directly manipulate the scheduling policy to change the participants' workload, adhere to or ignore the workflow preferences of the participants', and vary their level of control input into the algorithm. Further, requiring subjects to train and validate the robot's apprenticeship scheduling policy would be a limiting factor in conducting the experiment. As such, I chose to employ an optimization algorithm (i.e., Tercio), that allowed me to readily manipulate the desired parameters for a controlled experiment.

I now step through the Tercio algorithm. As shown in Figure 6-1, the algorithm takes as input a temporal constraint problem, a list of agent capabilities (i.e., the lowerbound, upperbound and expected duration for each agent performing each task) and the physical location of each task. Tercio first solves for an optimal task allocation by ensuring that the minimum amount of work assigned to any agent is as large as possible, as depicted in Equation 6.14. In this equation, $\boldsymbol{Agents}$ is the set of agents,

$A^a_{\tau_i^j}$ is a task allocation variable that equals 1 when agent a is assigned to subtask τ_i^j and 0 otherwise, $\boldsymbol{A}$ is the set of task allocation variables, $\boldsymbol{A}^*$ is the optimal task allocation and $C^a_{\tau_i^j}$ is the expected time it will take agent a to complete subtask τ_i^j.

$$\boldsymbol{A}^* = \min_{\{\boldsymbol{A}\}} \max_{\boldsymbol{Agents}} \sum_{\tau_i^j} A^a_{\tau_i^j} \times C^a_{\tau_i^j}, \forall a \in \boldsymbol{Agents} \tag{6.14}$$

After determining the optimal task allocation, $\boldsymbol{A}^*$, Tercio uses a fast sequencing subroutine to complete the schedule. The sequencer orders the tasks through simulation over time. Before each commitment is made, the sequencer conducts an analytical schedulability test to determine whether task τ_i can be scheduled at time t given prior scheduling commitments. If the test returns that this commitment can be made, the sequencer then orders τ_i and continues. If the schedulability test cannot guarantee commitment, the sequencer evaluates the next available task.

If the schedule, consisting of a task allocation and a sequence of tasks, does not satisfy a specified makespan, a second iteration is performed by finding the second-most optimal task allocation and the corresponding sequence. The process terminates when the user is satisfied with the schedule quality or when no better schedule can be found. In this experiment, Tercio was specified to run for 25 iterations and return the best schedule.

Tercio was employed because it allows for easy altering of task allocation within its task allocation subroutine. Here, I describe the specific Tercio alterations incorporated into each experiment. Note that only the task allocation subroutine within Tercio was modified for the three experiments; the sequencing subroutine remained unaltered.

6.4.1 Algorithm Modifications for Mixed-Initiative Scheduling

The situational awareness experiment sought to determine whether situational awareness degrades as a robotic agent is allowed greater autonomy over scheduling decisions. This experiment considered three conditions: autonomous, semi-autonomous, and manual control. Under the autonomous condition, the robotic teammate performed scheduling for the entire team; as such, the robot could use Tercio without modifications.

Under the semi-autonomous condition, in which the human participant decides which tasks he/she will perform and the robotic agent decides how to allocate the remaining tasks between itself and a human assistant, Tercio was required to consider the tasks allocated by the participant. After the participant specified which tasks he/she would perform, the experimenter provided these assignments to the robot, which encoded the allocation as an assignment to the decision variables. Specifically, Tercio set $A^{participant}_{\tau_i^j} = 1, A^{asst.}_{\tau_i^j} = 0, A^{robot}_{\tau_i^j} = 0$ for subtasks τ_i^j assigned to the participant, and $A^{participant}_{\tau_x^y} = 0$ for subtasks τ_x^y the participant did not assign to him/herself. Thus, the robot (via Tercio) only needed to solve for the allocation variables not already allocated by the participant.

Under the autonomous condition, the participant specified all task allocation assignments. As such, the robotic agent set $A^a_{\tau_i^j} = 1$ for all subtasks τ_i^j assigned to agent a, and $A^a_{\tau_x^y} = 0$ for all subtasks τ_x^y not assigned to agent a, for all agents a.

6.4.2 Algorithm Modifications for Scheduling with Preferences

This work focused on the effect of incorporating the preferences of human team members when generating a team's schedule. Preferences can exist in a variety of forms. For example, humans may have preferences about the duration of events (how long it takes to complete a given task) or the duration between events (the lowerbound or upperbound on the time between two tasks) [259]. In this investigation, preferences related to task types were considered – for example, a worker may prefer to complete a drilling task rather than a painting task. Such preferences can be included in the mathematical formulation in Equations 6.3 through 6.13 as an objective function term where one seeks to maximize the number of preferred tasks assigned to the participant, as shown in Equation 6.15. In this equation, the objective function term for maximizing preferences is balanced with the established criteria (i.e., function $g\left(\{A^a_{\tau_i^j}|\tau_i^j \in \boldsymbol{\tau}, a \in A\},\ \{J_{\langle \tau_i^j, \tau_x^y \rangle}|\tau_i^j, \tau_x^y \in \boldsymbol{\tau}\},\ \{s_{\tau_i^j}, f_{\tau_i^j}|\tau_i^j \in \boldsymbol{\tau}\}\right)$ from Equation 6.3) via a weighting parameter α.

$$
\begin{aligned}
\min z,\, z = \alpha \times g\Big(&\{A^a_{\tau_i^j}|\tau_i^j \in \boldsymbol{\tau}, a \in A\},\\
&\{J_{\langle \tau_i^j, \tau_x^y \rangle}|\tau_i^j, \tau_x^y \in \boldsymbol{\tau}\}, \{s_{\tau_i^j}, f_{\tau_i^j}|\tau_i^j \in \boldsymbol{\tau}\}\Big)\\
&- (1-\alpha) \times \left(\sum_{\tau_i^j \in \boldsymbol{\tau_{preferred}}} A^{participant}_{\tau_i^j} \right)
\end{aligned}
\tag{6.15}
$$

Alternatively, one could incorporate preferences as a set of constraints on enforcement of a minimum or maximum level of preferred work assigned to the participant, as shown in Equations 6.16 and 6.17. In these equations, k_{ub}^{pref} and k_{lb}^{pref} are upper- and lowerbounds on the number of preferred tasks allocated to the participant, and $k_{ub}^{pref^c}$ and $k_{lb}^{pref^c}$ are upper- and lowerbounds on the number of non-preferred tasks allocated to the participant.

$$
k_{lb}^{pref} \leq \sum_{\tau_i^j \in \boldsymbol{\tau_{pref}}} A^{participant}_{\tau_i^j} \leq k_{ub}^{pref} \tag{6.16}
$$

$$
k_{lb}^{pref^c} \leq \sum_{\tau_i^j \in \boldsymbol{\tau_{pref^c}}} A^{participant}_{\tau_i^j} \leq k_{ub}^{pref^c} \tag{6.17}
$$

For these experiments preferences were modeled as a set of constraints, which were added to Tercio's task allocation subroutine. For the purpose of human-subject experimentation, where one must control for confounders, this approach offers greater control over schedule content, as opposed to including a preference term within the objective function. The challenge of using an objective function model is in the need to tune one or more coefficients (e.g., α in Equation 6.15) in the objective function to balance the contribution of the schedule efficiency (i.e., makespan) with the importance of adhering to preferences. This tuning across a variety of participants is difficult and inconsistent.

For all three conditions, $k_{lb}^{pref} = k_{lb}^{pref^c} = 0$. Under the positive condition, participants could be assigned only one task that did not align with their preferences (i.e., $k_{ub}^{pref} = \infty$ and $k_{ub}^{pref^c} = 1$) – participants preferring to build could be assigned one fetching task at most, and vice versa. Under the negative condition, participants could be assigned a maximum of one task that aligned with their preferences (i.e., $k_{ub}^{pref} = 1$ and $k_{ub}^{pref^c} = \infty$) – for example, participants preferring to build could be assigned one build task at most. Under the neutral condition, Tercio's task allocation subroutine would run without alteration (i.e., $k_{ub}^{pref} = k_{ub}^{pref^c} = 1$, $\tau_{preferred} = \emptyset$).

Based on results from previous studies indicating the importance of team efficiency [87, 88], these experiments sought to control for the influence of schedule duration on team dynamics. For the experiment studying scheduling preferences, 50 iterations of Tercio were run for each participant under the positive, neutral, and negative parameter settings, generating a total of 150 schedules. A set of three schedules, one from each condition, was identified such that the makespans were approximately equal. (The workload of the individual agents was not controlled for.) The robot then used these schedules to schedule the team under the respective conditions.

6.4.3 Algorithm Modifications for Workload- and Scheduling Preference-based Constraints

In this experiment, I needed to control for makespan across all four conditions while varying the participants' workloads and the types of tasks they were assigned.

To control for the degree to which preferences were included in the schedule, I again added Equations 6.16 and 6.17 to Tercio's task allocation subroutine. Under conditions with high preference, all tasks assigned to the participant were preferred tasks (i.e., $k_{ub}^{pref} = \infty$ and $k_{ub}^{pref^c} = 0$); under conditions with low preference, all tasks assigned to the participant were non-preferred tasks (i.e., $k_{ub}^{pref} = 0$ and $k_{ub}^{pref^c} = \infty$). Under all conditions, I set $k_{lb}^{pref} = k_{lb}^{pref^c} = 0$.

To control for the utilization of the participant, I added an objective function term to Tercio's task allocation subroutine that minimized the absolute value of the difference between the desired utilization of the participant U^{target} and the actual utilization of the participant $\sum_{\tau_i^j \in \boldsymbol{\tau}} A_{\tau_i^j}^{participant} \times lb_{\tau_i^j}$. Since the absolute value function is nonlinear and cannot be handled by a linear program solver, I linearized the term in Equations 6.18 and 6.19:

$$z_{utility} \geq U^{target} - \sum_{\tau_i^j \in \boldsymbol{\tau}} A_{\tau_i^j}^{participant} \times lb_{\tau_i^j} \tag{6.18}$$

$$z_{utility} \geq -U^{target} + \sum_{\tau_i^j \in \boldsymbol{\tau}} A_{\tau_i^j}^{participant} \times lb_{\tau_i^j} \tag{6.19}$$

I generated schedules for each condition in three steps: First, I ran Tercio without any alterations to the task allocation subroutine for 100 iterations. Tercio works by iteratively generating task allocations and then sequencing the task set given the corresponding task allocation. Each iteration takes approximately one-third of a second. By running Tercio for several iterations, I allowed it to explore the search space so that it could then identify a candidate schedule with given characteristics (e.g., a specific degree of utilization of a particular agent). From these iterations, I recorded the median utilization U^{median} of the participant.

Next, I ran four additional sets of 100 iterations of Tercio – one set for each of the four conditions listed above. As before, I used Equations 6.16 and 6.17 to control for the degree to which the robot included the participant's preferences while scheduling. When the preference variable was set to high, I set $k_{ub}^{pref} = \infty$ and $k_{ub}^{pref^c} = 0$, and I set $k_{ub}^{pref} = 0$ and $k_{ub}^{pref^c} = \infty$ for the low preference condition. In both conditions, $k_{lb}^{pref} = k_{lb}^{pref^c} = 0$.

In the experiment studying workload, I controlled for the participant's utilization via Equations 6.18 and 6.19. When the utilization variable was set to high, I set $U^{target} = U^{median}$. When the utilization variable was set to low, I set $U^{target} = \frac{U^{median}}{2}$.

I then identified one schedule from each of the four sets of 100 Tercio iterations to generate a set of schedules with short, approximately equal makespans and utilizations close to their respective targets. To generate this set, I employed Equation 6.20, which minimizes the difference between the longest and shortest makespans across

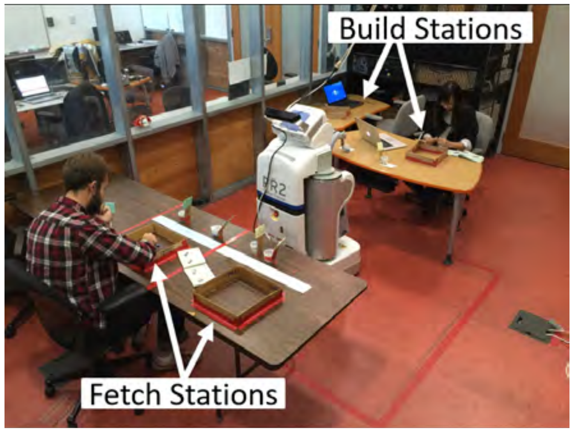

Figure 6-2: This figure depicts a diagram of the laboratory room where the experiment took place. There were two locations where the human and robot workers could inspect part kits during a fetching task, and two locations where the human workers built part kits.

the four conditions (i.e., $\max_{i,j}(m_i - m_j)$), the longest makespan (i.e., $\max_i m_i$) and the maximum difference between each schedule's target utilization U_i^{target} and its actual utilization U_i. In the experimental procedure, I set $\alpha_1 = \alpha_2 = 1, \alpha_3 = 2$.

$$z_{tuning} = \alpha_1 \max_{i,j \in schedules} (m_i - m_j) + \alpha_2 \max_{i \in schedules} m_i + \alpha_3 \max_{i \in schedules} \left(U_i^{target} - U_i\right) \tag{6.20}$$

6.5 Experimental Design

I conducted a series of three human-participant experiments ($n = 17$, $n = 18$, $n_3 = 20$) that required the fetching and assembly of Lego part kits. The goal of these experiments was to assess the following: 1) how a robotic teammate's inclusion of the preferences of its human teammates while scheduling affects team dynamics, 2) how the benefits of including these scheduling preferences varies as a function of the degree to which the robot utilizes the human participant, and 3) how situational awareness degrades as a function of the level of autonomy afforded to the robot over scheduling decisions. I used the same basic experimental setup for all three experiments, which I describe below.

6.5.1 Materials and Setup

My human-robot manufacturing team consisted of the human participant, a robotic assistant, and a human assistant. The human participant was capable of both fetching and building, while the robot assistant was only capable of fetching. One of the experimenters played the role of a third teammate (the human assistant) for all participants and was capable of both fetching and building. This human assistant was included in order to more realistically represent the composition of a human-robot team within a manufacturing setting. A Willow Garage PR2 platform, depicted in Figure 6-2, was used as the robotic assistant for the human-robot team. The robot used adaptive Monte Carlo localization (AMCL) [82] and the standard *Gmapping* package in the Robot Operating System (ROS) for navigation.

6.5.2 Procedure

The scenario included two types of tasks: fetching and assembling part kits. As shown in Figure 6-2, the experiment environment included two fetching stations and two build stations, with four part kits located at each fetching station.

Fetching a part kit required moving to one of two fetching stations where the kits were located, inspecting the part kit, and carrying it to the build area. The architecture of the fetching task is analogous to actions required in many manufacturing domains. In order to adhere to strict quality assurance standards, fetching a part kit required verification from one to two people that all of the correct parts were present in the kit, as well as certification from another person that the kit had been verified. Additional constraints were imposed in order to better mimic an assembly manufacturing environment: A part kit must have been fetched before it could be built, and no two agents were able to occupy the same fetching or build station at the same time.

Agents were required to take turns using the fetching stations, as allowing workers to sort through parts from multiple kits at the same location risked the participants mixing the wrong part with the wrong kit. Furthermore, in manufacturing, if a part or part kit is missing from an expected location for too long, work in that area of the factory will temporarily cease until the missing item has been found. As such, a 10-minute deadline was imposed from the time that the fetching of a part kit began until that kit had been built.

Assembly of the Lego model involved eight tasks $\boldsymbol{\tau} = \{\tau_1, \tau_2, \ldots, \tau_8\}$, each of which consisted of a *fetch* and *build* subtask $\tau_i = \{\tau_i^{fetch}, \tau_i^{build}\}$. The amount of time each participant took to complete each subtask $C_i^{participant-fetch}$ and $C_i^{participant-build}$ was measured during a training round. The timings for the robot $C_i^{robot-fetch}$ and human assistant $C_i^{assist-fetch}$ and $C_i^{assist-build}$ (performed by an experimenter) were collected prior to the experiments.

In all three experiments, the robotic agent employed Tercio as a dispatcher, communicating to the participant and human assistant when to initiate their next subtasks. Tercio would tell each agent when they were able to initiate or complete each subtask, and each agent would send a message acknowledging initiation or completion via simple, text-based messages over a TCP/IP GUI[2].

Modifications for the Experiment Studying Situational Awareness

For the study evaluating the effects of mixed-initiative scheduling on the situational awareness of the human team members, a between-participants experiment was performed in which each participant experienced only one of three conditions: autonomous, semi-autonomous, or manual.

As stated above, under the autonomous condition, the robot scheduled the three members of the team using Tercio with the default task allocation subroutine. Under the semi-autonomous condition, each participant selected which tasks they would perform and the robot allocated the remaining tasks to itself and the human assistant. Under the manual condition, the participant allocated tasks to each of the team members. The robot sequenced the tasks under all conditions.

After the human and/or robot completed the task allocation and sequencing

[2] SocketTest v3.0.0 ©2003-2008 Akshathnkumar Shetty (`http://sockettest.sourceforge.net/`)

process, the participants were allowed 3 minutes to review the schedule. In prior work [87], participants required approximately 3 minutes to perform task allocation; as such, participants were given at least this much time to review a robot-generated schedule under the autonomous condition. Participants were not told they would later respond to questionnaires about their experiences because I did not want to unduly bias them to focus on preparing for such a questionnaire. Instead, I wanted participants to fully attend to carrying out the task at hand.

After the participants reviewed the schedule, the team executed their tasks according to that schedule. At approximately 200 seconds into execution, the experimenter halted the process and administered the post-trial questionnaire (as shown in Table 6.1) according to the SAGAT technique. The timing of the intervention was tuned to allow each team member to have been assigned at least one task on average. The team did not complete the schedule after the SAGAT test; the experiment concluded following administration of the questionnaire

Extensions for the Experiment Studying Scheduling Preferences

For the experiment studying scheduling preferences, I employed a within-participants design. As such, participants experienced all three experimental conditions: positive, neutral, and negative. The order in which participants experienced these conditions was randomized. Participants were randomly assigned to these conditions. At the beginning of each condition, participants were told their robot teammate wanted to know whether they preferred to complete fetch tasks or build tasks, and the participants responded accordingly.

Deference to the participants with regard to their preferred tasks is in keeping with a pseudo-experiment. I did not attempt to balance participants according to the number in the sample who preferred fetching vs. building, as fourteen of eighteen participants (78%) preferred building tasks. Participants were not informed a priori of the different conditions; as such, subjective evaluations of team dynamics under each condition would not be influenced by the expectation that the robot would or would not cater to the participants' preferences.

The preferences, along with task completion times for each of the three team members, were provided to the robot, which scheduled the team. The team then performed the tasks to completion. After the schedule was completed, participants received the post-trial questionnaire depicted in Table 6.3. This process was repeated once for each condition, as indicated previously. After completing the tasks under all three conditions, the participant received the post-test questionnaire shown in Table 6.4. The experiment concluded after completion of this questionnaire.

Extensions for the Experiment Studying Workload

For the experiment studying workload influence, I employed an experimental design that mirrored the procedure for the experiment studying workflow preferences, with one primary difference: I varied workload and the degree to which human preferences were considered during scheduling, rather than preferences alone. Participants were not informed about whether the robot was varying their utilization, and the schedule itself was not reported to the participant; participants had to infer changes to their degree of utilization based only on their subjective experience.

6.6 Results

In this section, I report the results from statistical analysis of the experiments. Statistical significance is measured at the $\alpha = 0.05$ level.

6.6.1 Participants

I recruited participants for all three experiments from a local university. The sample population for the situational awareness study consisted of 20 participants (six men and 14 women) with an average age of 19.5 ± 1.95 years (range, 18 to 25 years). The cohort for the study of scheduling preferences included 18 participants (10 men and eight women) with an average age of 27 ± 7 years (range, 19 to 45 years). The cohort for the workload study consisted of 18 participants (10 men and eight women) with an average age of 21 ± 3 years (range, 18 to 30 years). In all experiments, participants were assigned to the various experimental conditions via random sampling without replacement, so as to balance participants across the conditions.

6.6.2 Results for Situational Awareness

Recall that the associated hypothesis **H1** states that human participants' situational awareness would decline as the robot's autonomy over scheduling decisions increased.

I administered a SAGAT-based test in which participants received a questionnaire consisting of both objective and subjective measures. I observed statistically significant decreases in situational awareness and participants' confidence in their situational awareness while under the autonomous condition, when the robot had full control over scheduling decisions.

Figure 6-3 depicts the team situational awareness score for Questions 1, 5, 9, and 13 from the post-trial questionnaire (shown in Table 6.1). For visual clarity when comparing the results from each question, I have normalized the values for each question in Figure 6-3 such that the maximum team score for each question is equal to 1.

I conducted a mixed-factor analysis of variance (ANOVA) for Question 1, and observed a statistically significant difference for participants' responses to Question 1 ($F(2, 17) = 3.894, p < 0.041$) across the three conditions. Results from a pair-wise comparison with a Student's t-test indicated that participants were statistically significantly more accurate when recalling which action team members performed under the semi-autonomous condition ($M = 0.67$,$SD = 0.48$) than the autonomous condition ($M = 2.13$,$SD = 1.36$), ($t(12), p < 0.014$). The manual condition ($M = 1.00$,$SD = 0.89$) was not statistically significantly different from the other two conditions.

I also applied a set of pair-wise t-tests with a Bonferroni correction $\alpha' = \frac{\alpha}{3} = \frac{0.05}{3} = 0.01\bar{6}$ for responses to Question 9, and found that participants were less accurate when recalling all previous actions of each agent under the autonomous condition ($M = 7.88$,$SD = 2.75$) compared with the manual condition ($M = 3.38$,$SD = 3.49$) ($p < 0.0158$). There was no statistically significant difference with regard to the

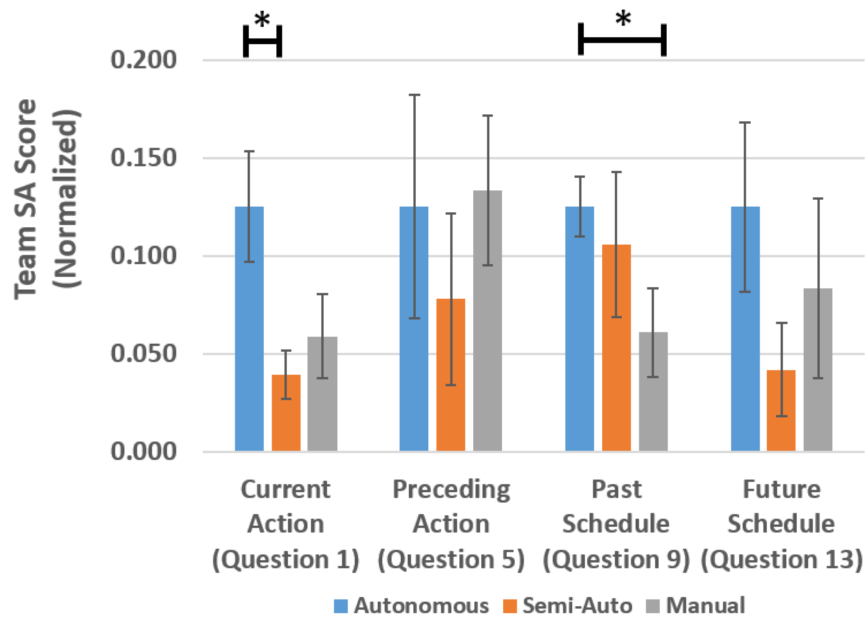

Figure 6-3: This figure depicts participants' average SA scores for Questions 1, 5, 9, and 13 in the post-trial questionnaire shown in Table 6.1, as a function of the degree of automation over scheduling decisions. The standard error of the mean is shown as whisker bars. Note that a lower score indicates a better situational awareness.

semi-autonomous ($M = 6.67, SD = 5.71$) condition.

Next, the participants' responses to the set of participant questions from Table 6.1 (i.e., Questions 2-4, 6-8, 10-12, and 14-16) are considered. Choosing the correct test was challenging. In general, a statistician would want a composite measure for confidence as it is more robust to false positives. Such a measure would combine responses to Questions 2-4, and likewise for Questions 6-8, 10-12, and 14-16, as a repeated measure. However, one could not immediately apply an ANOVA in this manner because the data were on an ordinal rather than an interval scale.

As such, two types of analysis were performed on these data. First, a non-parametric analysis was used, assuming ordinal, non-normally distributed data. In this analysis, the confidence of an individual participant under a given condition for the current actions of agents (referring to Questions 2-4), the median of the answers were used as a single data point (i.e., the median response across Questions 2-4). Second, a set of medians for participants under each condition was constructed and compared to the medians for other participants. For a qualitative description, a

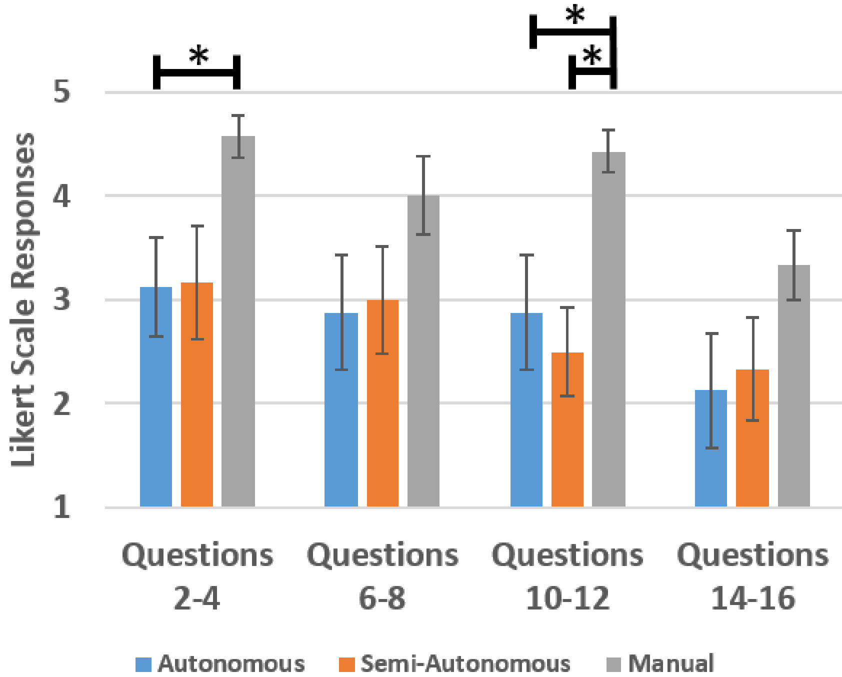

Figure 6-4: This figure depicts the average of the medians of participants' responses to Likert-response Questions 2-4, 6-8, 10-12, and 14-16 under the autonomous, semi-autonomous and manual conditions. The standard error of the mean is shown as whisker bars.

histogram of the medians is depicted in Figure 6-4.

Results from an omnibus Kruskal-Wallis test indicated significant differences across the conditions with regard to participants' confidence in their situational awareness for the current activities (Questions 2-4; $\chi^2(2) = 6.09$, $p = 0.0476$), and past activities of their team members (Questions 10-12; $\chi^2(2) = 7.98$, $p = 0.018$). A pair-wise Kruskal-Wallis test indicated that participants were statistically significantly more confident in their situational awareness for the current activities of team members (Questions 2-4) when under the manual condition than the semi-autonomous ($\chi^2(1) = 5.61$, $p < 0.018$) or autonomous ($\chi^2(1) = 4.04$, $p < 0.044$) conditions. Likewise, participants were found to be statistically significantly more confident in their situational awareness for the current activities of team members (Question 10-12) under the manual condition than the semi-autonomous ($\chi^2(1) = 7.93$, $p < 0.005$) or autonomous ($\chi^2(1) = 4.15$, $p = 0.0416$) conditions.

In the second analysis, the data were treated as interval data. Prior work has included extensive analyses suggesting that one can reasonably approximate a symmetric Likert-response format as interval data [37, 38], and that the F-test is quite robust with respect to breaking the assumptions of normality with regard to interval data [86]. A mixed-factor ANOVA was used to measure the composite confidence for the sets of questions corresponding to the current, preceding, past, and future actions of team members.

A mixed-factor ANOVA found that the level of robotic autonomy over scheduling decisions affected participants' confidence in their knowledge of the current actions of their team (Questions 2-4, $F(2, 18) = 4.228, p < 0.031$), as well as their confidence in their knowledge of the team's previous actions (Questions 6-8, $F(2, 18) = 6.293, p < 0.008$). These findings support the results from the Kruskal-Wallis test.

Upon performing pair-wise comparisons of the autonomous, semi-autonomous, and manual conditions using the mixed-factor ANOVA, there was statistically significantly greater confidence in situational awareness among participants with regard to the current activities of team members (Questions 2-4) under the manual condition than in the autonomous condition ($F(1, 13) = 11.377, p = 0.005$). Likewise, participants were statistically significantly more confident in their situational awareness about the current activities of team members (Questions 10-12) when under the manual condition than the semi-autonomous ($F(1, 11) = 18.615, p = 0.001$) or autonomous conditions ($F(1, 11) = 8.960, p = 0.010$). These findings corroborate those from non-parametric testing and strongly suggest that participants have less confidence in their situational awareness when under the autonomous condition.

6.6.3 Results for Scheduling Preferences

Recall that hypothesis **H2** states that human participants would prefer to work with a robot when it included their workflow preferences in scheduling decisions. Based on responses to Questions 22-24 in Table 6.4, there was a statistically significant evidence that human participants preferred working with a robot that included their preferences when scheduling ($p < 0.001$). Participants reported that they would rather work with a robotic teammate that included their preferences than one that

Table 6.6: Correlation between utilization and participants' perception of the team ($N = 17$).

Q.	Correlation Coefficient	t-value	p-value
7	$r = 0.285$	$t = 2.084$	$p = 0.021$
10	$r = 0.311$	$t = 2.287$	$p = 0.013$
14	$r = 0.286$	$t = 2.086$	$p = 0.021$
15	$r = 0.269$	$t = 1.957$	$p = 0.028$

was unaware of their preferences ($p < 0.001$). Furthermore, participants reported that they would prefer to work with a robot that was unaware of their preferences than a robot that scheduled according to the opposite of their preferences ($p < 0.001$). Participants felt the robot liked them more (Question 6 in Table 6.4) under the neutral condition, when the robot was unaware of the participants' preferences, than under the negative condition ($p < 0.05$). These results support the hypothesis that the preferences of human workers are important for a robotic teammate to include when making scheduling decisions.

Surprisingly, the amount of work allocated to participants had a strong impact on their subjective perceptions of their teams' interactions. In post-hoc analysis, the Pearson product-moment correlation coefficient was calculated for the rank of participants' Likert-scale responses to questions from the post-trial questionnaire (Table 6.3) for each condition as a function of the amount of work assigned to each participant; a statistically significant proportion of Likert-Scale responses responses on 16 of the 21 questions[3] were positively correlated with the amount of time assigned to participants ($\chi^2 = 5.762$, $p = 0.016$). Furthermore, four of the 16 with a positive correlation (Questions 7, 10, 14, and 15 from Table 6.3) were statistically significantly correlated, as shown in Table 6.6 ($p < 0.05$). There did not exist a statistically significant negative correlation with the amount of time assigned to participants.

To further investigate this finding, an analysis of variance found that the robot allocated a statistically significantly different amount of work to the participant as a function of how the robot included the participant's preferences when scheduling, as shown in Figure 6-5 (ANOVA $F(2, 48) = 5.16, p = 0.009$). Interestingly, participants

[3]Questions 1-7, 9-10, 12-16, 18, and 20 from Table 6.3 showed participants' responses were positively correlated with their utilization.

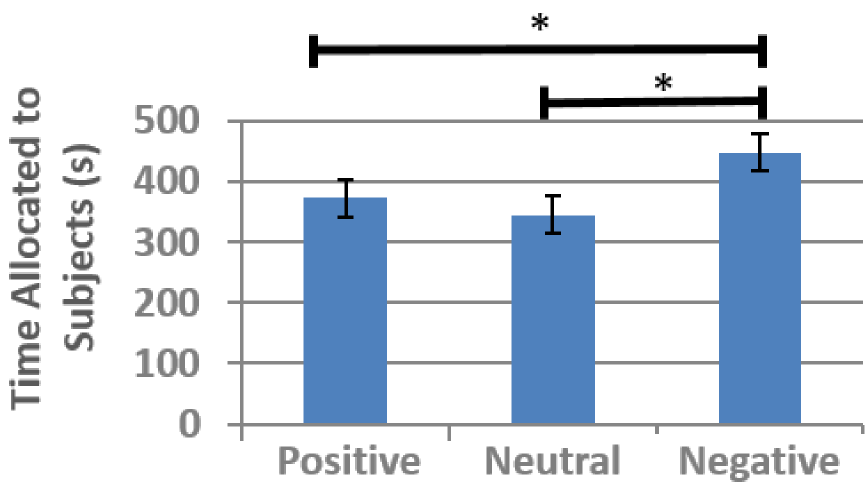

Figure 6-5: This figure depicts the mean and standard error of the amount of work, in seconds, assigned to the participant by the robotic teammate. Horizontal bars with an asterisk denote statistical significance ($p < 0.01$).

were allocated statistically significantly more work, as measured in seconds, when under the negative condition (M= 448, SD= 113) compared with the positive (M= 373, SD= 92) ($t(16) = 1.86$, $p = 0.04$) or neutral conditions (M= 345, SD= 82) ($t(17) = 2.14$, $p = 0.03$).

In collecting participants' preferences for the types of tasks they would rather complete, fourteen of eighteen participants (78%) reported they preferred to build the part kits rather than fetch them. Under the positive condition, participants received a maximum of one fetching task; under the negative condition, they received a maximum of one building task. The third teammate (the human assistant) was typically more proficient at building than the average participant; consequently, the optimal work allocation would typically assign the majority of building tasks to the assistant, with the participant providing support by performing more fetching tasks. (The robot teammate was only able to fetch part kits.) As such, the negative condition afforded participants the opportunity to complete a larger share of the work. Based on this result, I propose that participants' preferences for task types must be balanced with an innate desire on the part of a human worker to be an important contributor to his or her team.

6.6.4 Results of Varying Workload

Recall that **H3.A** states that participants would prefer to work with a robotic agent that included their workflow preferences, and **H3.B** states that participants would prefer working with a robotic agent that provided them with a relatively high workload. To test the hypotheses, an experiment was conducted to control for both the degree to which preferences were included and the degree to which participants were utilized.

Across 12 measures of the participants' perceptions about the human-robot team, there is statistically significant evidence that participants preferred working with a robot that included their preferences when scheduling – and, across 10 measures, that participants preferred working with a robot that utilized them more frequently, as opposed to working with a robot that is antagonistic to their preferences. Results are depicted in Table 6.7. For the findings reported here, an omnibus Friedman test was first used to determine that a statistically significant difference existed across all conditions, and then applied a pair-wise Friedman test to examine the differences between the conditions.

The experiment in which a robotic agent included the preferences of a human team member when scheduling provided initial evidence that human participants would prefer to work with a robotic agent when it considered their workflow preferences and more frequently utilized their time. These data statistically significantly support the hypotheses: When a robot schedules for a human-robot team, the human team members' perception of the robot and the team as a whole are significantly improved when the robot considers the preferences of the human worker and utilizes more of the workers' time.

In addition to these findings, there is a surprising trend between preferences, utilization and the participants' perception of team efficiency in post-hoc analysis. Under the high preference - high utilization condition, participants felt more strongly that the team performed tasks using the least possible amount of time, even though the schedule duration (i.e., makespan) was constant across all trials within participants

Table 6.7: P-values for statistically significant post-trial questions ($N = 18$). Statistically significant values are bolded.

Q.	Omnibus	High Util. vs. Low Util.	High Pref. vs. Low Pref.
2	$\boldsymbol{p = 0.013}$	$\boldsymbol{p = 0.002}$	$p = 0.096$
5	$\boldsymbol{p = 0.010}$	$\boldsymbol{p = 0.003}$	$\boldsymbol{p = 0.020}$
7	$\boldsymbol{p = 0.026}$	$\boldsymbol{p = 0.035}$	$\boldsymbol{p = 0.016}$
9	$\boldsymbol{p < 0.001}$	$\boldsymbol{p < 0.001}$	$p = 0.170$
10	$\boldsymbol{p < 0.001}$	$\boldsymbol{p = 0.007}$	$p = 0.061$
11	$\boldsymbol{p = 0.026}$	$\boldsymbol{p = 0.027}$	$\boldsymbol{p = 0.029}$
13	$\boldsymbol{p < 0.001}$	$\boldsymbol{p = 0.001}$	$\boldsymbol{p = 0.001}$
14	$\boldsymbol{p < 0.001}$	$\boldsymbol{p < 0.001}$	$\boldsymbol{p = 0.004}$
15	$\boldsymbol{p < 0.001}$	$\boldsymbol{p = 0.005}$	$\boldsymbol{p = 0.001}$
16	$\boldsymbol{p = 0.010}$	$\boldsymbol{p = 0.011}$	$\boldsymbol{p = 0.003}$
17	$\boldsymbol{p < 0.001}$	$\boldsymbol{p = 0.011}$	$\boldsymbol{p < 0.001}$
18	$\boldsymbol{p = 0.004}$	$\boldsymbol{p = 0.012}$	$\boldsymbol{p = 0.012}$
20	$\boldsymbol{p = 0.026}$	$p = 0.052$	$\boldsymbol{p = 0.013}$

($p < 0.004$). In the interests of further investigation, I propose a follow-on study examining how human team members' perceptions of the passage of time and team efficiency is affected by the way in which a robot schedules the team.

6.7 Discussion

6.7.1 Design Guidance for Roboticists

This chapter investigated key gaps in prior literature by assessing how situational awareness is affected by the level of autonomy in mixed-initiative scheduling for human robot teams, the effects of increased or decreased workload in human-robot team fluency, and the role of workflow preferences in robotic scheduling. These findings can serve as the basis for design guidance for roboticists developing intelligent collaborative robots that engage in mixed-initiative decision-making with humans.

Human situational awareness is poorer when the robotic agent has full autonomy over scheduling decisions, as assessed by both objective and subjective measures. However, prior work has indicated that decreasing robotic autonomy over scheduling decisions reduces efficiency and decreases the desire of the human to work with a robotic agent. Therefore, the positive and negative effects of increasing the robot's role in decision making must be carefully weighed. If there is a high probability the human agent will have to intervene in order to adjust work allocations, or the potential cost of poorer human performance due to reduced situational awareness is high, then I recommend that the human retain primary decision making authority. If human intervention is unlikely, or the cost of poorer human performance is low, then the benefits of improved team efficiency can be safely achieved by allowing the robot to retain primary decision making authority. In many applications, a mixed-initiative approach in which the participant and robot collaborate to make decisions offers a suitable middle ground between the two ends of this spectrum.

Also, a human's perception of a robotic teammate scheduling a team's activities may improve when the human is scheduled to complete tasks that he or she prefers. However, human team members' perception of the robot may be negatively impacted when they are scheduled to be idle for much of the time. Providing human team members with more highly preferred tasks at the cost of decreasing the total amount of work assigned to them may, in fact, have more of a negative impact than assigning human team members less-preferred tasks. Although the degree to which these variables interact is likely to be application-specific, it cannot be assumed that increasing one criterion at the cost of the other will improve team fluency.

Collaborations with robots that participate in decision making related to the planning and scheduling of work present unique challenges with regard to preserving human situational awareness and optimizing workload allocation to human teammates while also respective their workflow preferences. Careful consideration is necessary in order to design intelligent collaborative robots that effectively balance the benefits and detriments of maintaining an increased role in the decision making process.

6.7.2 Limitations and Future Work

There are limitations to the findings of the experiments presented in this chapter. The sample population consisted of young adults enrolled from a local university campus, whereas the target population consists of older, working adults in the fields of manufacturing and search-and-rescue, among other domains. Impressions of robotic teammates, in general, may differ significantly between these populations.

Workers may also use different criteria to evaluate a human-robot team. For example, if chronic fatigue is an issue in a given setting, workers may prefer a greater amount of idle time. Also, the expression of preferences was limited to a binary choice between two types of tasks; however, the preferences of real workers may be more nuanced and difficult to encode computationally. For these reasons, I recommend a follow-on study, conducted in multiple factories across a variety of industries and work environments, in order to confirm the results of the experiments.

These experiments studied one robot form factor (i.e., a PR2) in the investigation. It is possible that other form factors could elicit a different response from participants. Further, a specific scheduling technique, Tercio, well-suited for human-robot teaming, was used. It is possible that alternate scheduling algorithms could alter the participants' experience.

When manipulating the degree to which participants are utilized and the amount of preferred work assigned to those participants, the experiment used "high" and "low" settings. Increasing the setting of these independent variables from low to high was found to have positively affected the participants' experience working with the robot. It is possible, however, that the relationship between utilization and participants' subjective experience is not linear. For example, an "extremely high" utilization could be less desirable than even low utilization. Future work should investigate utilization and workflow preferences across the entire spectrum.

Regarding The Factor Level Extremes in the Experiment on Workflow Preferences

Further, it is important to discuss the motivation for using opposing conditions in the experiment studying workflow preferences and utilization. This particular experiment considered two independent variables (i.e., factors): the degree to which workflow preferences were included and the degree to which the participants were utilized. Each variable had two factor levels: high and low. For the participants' utilization, the robot either scheduled them with more or less work. For the participants' workflow preferences, the robot either scheduled in favor of their preferences or opposite of their preferences (i.e., antagonistically); there was no condition for which the preferences were ignored.

However, in the first experiment studying workflow preferences, there were three factor levels: Positive, Neutral, and Negative. These factor levels corresponded to the robot incorporating the participants' preferences, ignoring those preferences, and scheduling opposite of those preferences. However, the results did not show definitely that the degree to which preferences were incorporated affected the participants' subjective perception of the team's fluency. Rather, the results showed an interplay between the degree to which preferences were incorporated and the utilization level of the participants in the experiment.

To tease out the relative effects of workflow preference inclusion and participants' utilization in this experiment, it was necessary to vary them separately as independent variables. Further, to show that there is *any* effect, it is only necessary to show that the independent variables, set at their extrema, result in a different experimental outcome. Using factor levels at the variables extrema to find a difference, however, does not provide one with an understanding of the nature of the gradation between those extrema – only that such a gradation exists.

The aim of the first experiment studying workflow preferences was to determine whether a roboticist should consider workflow preferences because it is possible that the way in which a robot incorporates those preferences might affect the team's interaction. The experimental results reported in this chapter support this consideration of workflow preferences – a robot that schedules antagonistically to a participant's preferences are less-desired by participants than a robot that schedules in favor to those preferences. However, the experimental result does not precisely explore of how including some preferences, ignoring all preference, or scheduling opposite of the preferences might affect the interaction. I recommend that a roboticist with a particular application in mind conduct an experiment that considers a fuller spectrum of factor levels.

A Word on Situational Awareness and Memory

Finally, I want to offer a word on a potential confounding factor for the investigation into how decision-making authority affects situational awareness. The aim of my experimental design was to test the hypothesis that having participants take a lesser role over scheduling decisions would decrease their situational awareness. Using the SAGAT technique [74], I found that participants were, in fact, less situationally aware of the human-robot teams activities during the execution of the teams schedule.

Recall from our discussion in Chapter 2 that Endsley defined a three-level model for situational Awareness [75]: perception (Level 1 SA), comprehension (Level 2 SA), and projection (Level 3 SA). These levels require the operator of a complex system to perceive the state of the environment, understand the meaning of this state, and project the state into the future to understand how that state must change [75].

The SAGAT questionnaire I employed in the experiment was aimed to gauge how well participants were able to perceive and be aware of the team's activities. I found that, by having the participant construct part (i.e., in the semi-autonomous condition) or all of the schedule (i.e., in the manual condition), participants were better able to complete the SAGAT questionnaire. This finding provides evidence that giving a robot full authority over scheduling decisions (i.e., in the autonomous condition), even if participants are allowed to review those decisions, decreases participants' situational awareness.

However, one could argue that, rather than participants' having a heightened ability to perceive in the manual or semi-autonomous conditions, they merely were recalling information for the questionnaire that they memorized at the beginning of the trial. In other words, constructing a schedule may be a better memorization tool than studying an already-constructed schedule which was prepared by the robot.

Yet, carrying this thought process one further step, better memorization may have helped participants develop better mental models of the team's activities. In fact, operators relying on their internalized mental models to supplement situational awareness is an integral component of the fields theoretical understanding of situational awareness Endsley [76].

Endsley hypothesized and developed evidence to support the notion that memory stored in the form of mental models or schema play a major role in helping people overcome the limitations to their own working memory. Specifically, these models help people more efficiently direct their attention to various features of their environment to maintain a sufficiently accurate awareness of that environment [73, 78]. The Endsley hypothesis [76] is supported by the works of many notable researchers in human factors [83, 128, 222].

Thus, one could argue that situational awareness and memory go hand in hand. In trials where the participants constructed their team's schedule, those participants might have relied on their memories in the form of mental models to help them more efficiently attend to the team's activities. Alternatively, having participants construct the schedule could have better engaged participants' attention to the team's activities (i.e., enhanced their perception). Regardless, requiring participants to construct part or all of the team's schedule did improve their ability to recall information, and, that information could be valuable if the participant were required to re-schedule. Thus, I still recommend that roboticists do not fully automate the scheduling decisions unless safeguards are developed to counteract human team members' lack of awareness of the team's activities.

6.8 Conclusions

While new computational methods have significantly enhanced the ability of people and robots to work flexibly together, there has been little study into the ways in which human factors must influence the design of these computational techniques. This chapter investigated how situational awareness varies as a function of the degree of autonomy a robotic agent has during scheduling, and found that human participants' awareness of their teams' actions decreased as the degree of robot autonomy increased. This indicates that the desire for increased autonomy and accompanying performance improvements must balanced with the risk for – and cost resulting from – reduced situational awareness. This chapter also studied how team fluency varies as a function of the workload given to a human team member by a robotic agent, and the manner in which a robot should include the workflow preferences of its human teammates in the decision making process. Results indicate a complex relationship between preferences, utilization, and the participants' perception of team efficiency. The three study results provide guidelines for the development of intelligent collaborative robots, and a framework for weighing the positive and negative effects of increasing the robot's role in decision making.

Chapter 7

Contribution and Future Work

7.1 Thesis Contribution

I envision a future where service robots are intelligent, adaptive teammates that can learn from and adapt to their human counterparts. Currently, these robots are explicitly tasked, programmed, and supervised, which places an undue burden on their human teammates. To move from a paradigm where these systems are micro-managed to one where robots can learn to adapt to their human teammates, we need new computational methods for learning from scheduling demonstration. Further, we need to understand the human factors design principles necessary to embody these computational techniques in physical systems. In this thesis, I have contributed to realizing this vision by developing an embodied apprenticeship scheduling framework and exploring human-robot interaction with my system.

I developed a novel machine-learning optimization technique, COVAS, which enables robots to learn how to coordinate team activities from observing human domain experts. COVAS relies on a new policy learning formulation, apprenticeship scheduling. The key to my policy learning approach was the use of pairwise comparisons between the features describing the action taken by the expert versus the features describing each action not taken, and vice versa. Through these comparisons, I constructed a classifier, $f_{priority}(\tau_i, \tau_j)$, which can be used to predict the action τ_i^* the algorithm thinks is most likely to be taken by the expert. COVAS uses this classifier to construct a schedule, which then serves as the input to COVAS' mathematical optimization subroutine. This optimization leverages the apprentice scheduler's solution to inform a mathematical optimization with a tight lowerbound and initial seed solution that can be used to more efficiently explore the search space. A key contribution is COVAS' ability to learn from good, but imperfect, human demonstration to more efficiently and optimally solve scheduling problems.

I validated this apprenticeship scheduling formulation in a synthetic data set as well as with real-world data from human decision-makers in ASMD and healthcare. Within ASMD, I showed apprenticeship scheduling can learn policies from expert human demonstration that are able to outperform the average human expert ($p < 0.011$). Further, COVAS is able to apply an apprentice scheduler trained on good, but imperfect, demonstrations of experts solving easier ASMD scenarios to optimally solve harder problems and do so an order of magnitude faster than state-of-the-art optimization techniques. I also showed that apprenticeship scheduling can learn from demonstration by resource nurses on a labor and delivery ward to generate scheduling advice accepted 90% of the time by nurses and physicians.

Having developed a framework from human demonstration for learning how to perform team coordination, I next investigated how to design a service robot from a human factors perspective. First, I studied how embodiment of a scheduling algorithm affects human teammates' trust and reliance on that algorithm. I demonstrated through human-subject experimentation that professionals working with the embodied system were better able to discern the quality of the advice generated by the system as opposed to when the system was un-embodied ($p < 0.05$). Next, I demonstrated that a humans' situational awareness decreases as a robotic agent assumes more responsibility for scheduling decisions ($p < 0.05$). Moreover, I showed that human participants desire to contribute to the team with a non-nominal workload and by performing tasks they prefer. However, workload and workflow preferences are tightly related: lowering workload for the sake of giving participants their desired tasks does not necessarily improve team fluency, and vice versa. Through this investigation, I developed design guidelines for developing and deploying autonomous service robots that contribute to scheduling decisions.

7.2 Recommended Future Work

My thesis serves as a strong contribution to the goal of enabling service robots to be intelligent teammates that can learn as apprentices rather than be explicitly tasked and controlled. Yet, not all problems in any field are solved in only one thesis. In this section, I propose future research directions that build upon my thesis to advance the capabilities of apprenticeship scheduling.

7.2.1 Learning to Teach

In my thesis, I explored the challenge of developing an apprenticeship scheduling technique that scales beyond the power of the single expert. I have shown my techniques can learn from a single expert how to optimally solve scheduling problems more complex than those demonstrated by the expert and do so an order of magnitude faster than state-of-the-art techniques. By replicating this learning system across multiple physical platforms, a single expert can simultaneously contribute to solving many scheduling problems.

Yet, there is another way to interpret the phrase, "scaling beyond the power of the single expert." Experts have an inherent responsibility to bestow their knowledge on others in their domain (i.e., to teach). I propose the challenge of using apprenticeship scheduling to teach *human* students. There are many interesting challenges that arise from this proposal. For example, it is likely that merely showing demonstrations of solved problems is helpful but not sufficient for transferring *knowledge*. Students may be able to mimic the decision-making with sufficient effort, but it is likely that students will be unable to understand *why* those decisions are being made. Being able to justify why one action was made over another would be impactful, not just for teaching but also for building confidence in the system.

Another interesting challenge arises when one compares how a machine might teach as opposed to how a human educator teaches. The machine might generate a set of example trajectories for students to mimic; however, the human educator's examples might be able to also convey latent information the students could capture in their studies. The manner in which a human generates a schedule may differ from the way a human would demonstrate to another human *how* to schedule. Thus, the policy for autonomous scheduling may need to be inherently different from a policy for autonomous teaching.

7.2.2 Learning with Similarities and Differences

Commonly, supervisory learning approaches (e.g., apprenticeship learning) assume the demonstrators provide a homogeneous set of demonstrations. In other words, the demonstrators must attempt to achieve the same goal in the same manner. However, this is not always true. For example, Sammut et al. showed that commercial pilots, tasked with executing the same flight plan, generate demonstrations so different that it is more effective to learn an individual policy for each individual pilot rather than aggregating their data [220].

Learning to organize data according to differences within that data is an inherently unsupervised learning problem (e.g., clustering demonstrators according to their behavior). There have been some works that have attempted to bring together supervised and unsupervised learning, such as the work by Nikolaidis et al. [186]. However, these methods often solve the problem by first clustering over the types of demonstrators to create homogeneous sets of demonstrations, and, second, applying separate supervisory learning instances on each homogeneous set. Such approaches are a vast improvement over learning individual policies for each person because the amount of data to learn from increases by a factor of $\frac{n}{k}$, where n is the number of demonstrations and k is the number of clusters. Yet, the approach is still learning from only $\frac{1}{k}^{th}$ of the total amount of data available.

I think it is important we develop new techniques that integrate unsupervised and supervised learning in a combined learning step, rather than as a hierarchical approach. Such a learning algorithm would have the advantage of being able to generalize across all demonstrators' similarities while still tailoring its actions for the individual differences among the demonstrators. A combined mathematical function, however, would be more computationally complex than the individual constituents of supervised and unsupervised learning, which would likely necessitate new approximation techniques to more efficiently solve the problem.

7.2.3 Robots that Support Rather than Degrade SA

Autonomous robots have the ability to enhance our capabilities, efficiency, and work environments. However, as these robots assume responsibility for more and more of the decision-making, humans will naturally fall further out-of-the-loop. Human factors research, including that found in this thesis, has frequently shown this effect to exist. Further, this effect can be fatal, particularly in safety-critical domains such as healthcare and military operations. Situational awareness is necessary because, without it, humans may misinterpret a robot's actions or be incapable of adequately assuming responsibility for the robot's task in the event of a robot failure.

The challenge posed to researchers is then to develop artificial intelligence methods that enable robots to assume responsibility without sacrificing the human team members' situational awareness. This challenge of developing human-machine interfaces that support situational awareness is not new [52, 55]. These works typically focus on a scenario where a human must supervise a team of unmanned vehicles, and the key challenge is in designing clever user interfaces that convey information appropriately. However, enabling *robots* as physical team members to communicate their awareness to humans is relatively unexplored.

Recent work by Unhelkar et al. has posed algorithms that enable robots to intuit what information to convey, and when, to other robotic team members based on their model of the other team members' decision-making process [250]. This approach, in particular, assumes the robotic agents operate according to a Markov decision process. By understanding how an agent would change its behavior after receiving new information, one can decide whether or not to communicate that information.

However, developing and integrating a model of human decision-making, which is unlikely to be strictly Markovian, is an open question. A technique that could capture and reason about models of human decision-making could be beneficial in creating robots to enable human teammates to maintain their situational awareness.

7.2.4 Long-term Interactions

Many studies in human-robot interaction are cross-sectional, rather than longitudinal. This thesis, for example, conducted a user study in a hospital in which nurses and physicians work with a robot for approximately one hour. While I found significant positive effects by embedding apprenticeship scheduling in a robot over using an un-embodied system, it is possible that this effect would diminish (e.g., there may be a novelty effect that wears off), or even increase, if the interaction between the participant and the robot were to continue for days, weeks, or years. Such transient effects can be difficult to tease out without a significant time investment. Nonetheless, if these systems are to transfer from the research lab to the hospital, factory, and battlefield, it is critical that researchers undertake longitudinal studies of these systems.

Yet, longitudinal interactions involve more than merely human factors considerations. There are also important technical challenges. If these robotic systems are meant to learn from human demonstration, a human operator or teammate would expect that system to *continue* learning. As a first step, one could embed apprenticeship learning in a robotic platform and, at the conclusion of each interaction, have the algorithm re-train its policy given the old and newly acquired information.

This initial approach may be effective, but there could be some challenges. For example, human operators would have little patience for multiple errors of omission or commission on the part of the robot. Once the human operator offers correction, the robot's designer may want the robot to weigh the new and contrary example heavily to demonstrate to the human that the robot has incorporated this new information.

However, heavily weighing new information could result in altering parts of the robot's policy that were learned correctly the first time. Thus, it may be better to have a blend between a robust, validated policy and a database of exceptions for use in case-based reasoning. Exploring the intersection of generalization and specific case-based reasoning would be fruitful for future work.

7.2.5 Further Areas of Consideration

In addition to the open questions posed above, I provide the following areas for consideration.

- *Is there a general mathematical description of problems in the complex-dependencies (CD) domain?* Whereas current work on problems with CD are ad hoc in nature, developing a common basis for these problems could improve theoretical advances in the field.

- *How can we exactly model tasks that can be decomposed without enumerating all possible decompositions?* Consider a scenario where, depending on which robots work on a task, the nature of the task (e.g., its duration or location) changes. Without knowing a priori when one robot will stop working on that task and when a second robot will begin, one must enumerate all possible ways the task can be decomposed. Then, the optimization algorithm would search through the decompositions to find the best. However, this exponentially increases the search space. Rather, I ask, is there a way to compactly represent a task and its decompositions exactly without enumerating all possible decompositions?

- *Are there alternative formulations for exact discrete apprenticeship learning?* Apprenticeship learning typically relies on using regression to learn an objective function. In the case of IRL, the algorithm then uses reinforcement learning to generate an optimal policy. In the case of PTIME, a MILP is used to optimally solve for a specific problem instance [19]. While both are optimal, both are intractable for large problem sizes. In reinforcement learning, there are an exponential number of states to explore. In MILP, there are an exponential number of solutions to consider. I put forth as a challenge the goal of formulating an alternative to reinforcement learning and MILP that is exact and computationally efficient.

Bibliography

[1] CMU sphinx open source speech recognition toolkit, January 2016.

[2] Pieter Abbeel and Andrew Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *Proc. ICML*, 2004.

[3] Julie A. Adams. Multiple robot-single human interaction: effects on perceived workload and performance. *Behavior and Information Technology*, 28(2):183–298, 2009.

[4] Rachid Alami, Raja Chatila, Aurélie Clodic, Sara Fleury, Matthieu Herrb, Vincent Montreuil, and Emrah Akin Sisbot. Towards human-aware cognitive robots. In *The Fifth International Cognitive Robotics Workshop (The AAAI-06 Workshop on Cognitive Robotics)*, 2006.

[5] Jacopo Aleotti and Stefano Caselli. Robust trajectory learning and approximation for robot programming by demonstration. *Robotics and Autonomous Systems*, 54(5):409–413, 2006.

[6] Jennifer Alsever. Robot workers take over warehouses. *CNN Money*. `http://money.cnn.com/2011/11/09/smallbusiness/kiva_robots/`, November 9, 2011.

[7] Ulrich Anders and Olaf Korn. Model selection in neural networks. *Neural Networks*, 12(2):309 – 323, 1999.

[8] L. Ardissono, G. Petrone, G. Torta, and M. Segnan. Mixed-initiative scheduling of tasks in user collaboration. In *Proc. Eighth International Conference on Web Information Systems and Technologies*, pages 342–351, 2012.

[9] Wilma A Bainbridge, Justin Hart, Elizabeth S Kim, and Brian Scassellati. The effect of presence on human-robot interaction. In *In Proc. International Symposium on Robot and Human Interactive Communication (RO-MAN)*, pages 701–706. IEEE, 2008.

[10] Wilma A Bainbridge, Justin W Hart, Elizabeth S Kim, and Brian Scassellati. The benefits of interactions with physically present robots over video-displayed agents. *International Journal of Social Robotics*, 3(1):41–52, 2011.

[11] Ashis Gopal Banerjee, Masahiro Ono, Nicholas Roy, and Brian Williams. Regression-based lp solver for chance-constrained finite horizon optimal control with nonconvex constraints. pages 131–138, June 2011.

[12] Jimmy Baraglia, Maya Cakmak, Yukie Nagai, Rajesh Rao, and Minoru Asada. Initiative in robot assistance during collaborative task execution. In *2016 11th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 67–74. IEEE, 2016.

[13] Michael J. Barnes, Jessie Y. C. Chen, Florian Jentsch, and Elizabeth S. Redden. Designing effective soldier-robot teams in complex environments: training, interfaces, and individual differences. In *Proc. International Conference on Engineering Psychology and Cognitive Ergonomics (EPCE)*, pages 484–493. Springer, 2011.

[14] Andrew G. Barto and Sridhar Mahadevan. Recent advances in hierarchical reinforcement learning. *Discrete Event Dynamic Systems*, 13(1-2):41–77, 2003.

[15] Chumki Basu, Haym Hirsh, and William Cohen. Recommendation as classification: Using social and content-based information in recommendation. In *Proc. Fifteenth National Conference on Artificial Intelligence*, pages 714–720. AAAI Press, 1998.

[16] Randal W. Beard, Timothy W. McLain, Michael A. Goodrich, and Erik P. Anderson. Coordinated target assignment and intercept for unmanned air vehicles. *IEEE Transactions on Robotics and Automation*, 18(6):911–922, 2002.

[17] Jacques F. Benders. Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik*, 4:238–252, 1962.

[18] Pauline Berry, Bart Peintner, Ken Conley, Melinda Gervasio, Tomás Uribe, and Neil Yorke-Smith. Deploying a personalized time management agent. In *Proc. AAMAS*, pages 1564–1571, 2006.

[19] Pauline M. Berry, Melinda Gervasio, Bart Peintner, and Neil Yorke-Smith. Ptime: Personalized assistance for calendaring. *ACM Trans. Intell. Syst. Technol.*, 2(4):40:1–40:22, July 2011.

[20] Demitri P. Bertsekas. A distributed algorithm for the assignment problem. Technical report, Cambridge, Massachusetts Institute of Technology, Laboratory for Information and Decision Systems (LIDS), 1979.

[21] Demitri P. Bertsekas. Auction algorithms for network flow problems: A tutorial introduction. *Computational Optimization and Applications*, 1:7–66, 1990.

[22] Dimitris Bertsimas and Robert Weismantel. *Optimization over Integers*. Dynamic Ideas, Belmont, 2005.

[23] Elizabeth Blickensderfer, Janis A. Cannon-Bowers, and Eduardo Salas. Cross-training and team performance. In *Making decisions under stress: Implications for individual and team training*, pages 299–311. American Psychological Association, Washington, DC, 1998.

[24] Richard Bloss. Mobile hospital robots cure numerous logistic needs. *Industrial Robot: An International Journal*, 38(6):567–571, 2011.

[25] Craig Boutilier, Ronen I. Brafman, Carmel Domshlak, Holger H. Hoos, and David Pool. Cp-nets: A tool for representing and reasoning with conditional ceteris paribus preference statements. *Journal of Artificial Intelligence Research*, 21:135–191, February 2004.

[26] Craig Boutilier, Ronen I. Brafman, Holger H. Hoos, and David Poole. Reasoning with conditional ceteris paribus preference statements. In *Proc. UAI*, UAI'99, pages 71–80, 1999.

[27] Steven J. Bradtke and Michael O. Duff. Reinforcement learning methods for continuous-time markov decision problems. In *NIPS*, pages 393–400. MIT Press, 1994.

[28] Lisa Brandenburg, Patricia Gabow, Glenn Steele, John Toussaint, and Bernard J. Tyson. Innovation and best practices in health care scheduling. Technical report, February 2015.

[29] Jeffrey B. Brookings, Glenn F. Wilson, and Carolyne R. Swain. Psychophysiological responses to changes in workload during simulated air traffic control. *Biological Psychology*, 42(3):361 – 377, 1996. Psychophysiology of Workload.

[30] Peter Brucker. *Scheduling Algorithms*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 3rd edition, 2001.

[31] Luc Brunet, Han-Lim Choi, and Jonathan P. How. Consensus-based auction approaches for decentralized task assignment. In *Proc. AIAA Guidance, Navigation, and Control Conference (GNC)*, Honolulu, HI, 2008.

[32] Luc Brunet, Han-Lim Choi, and Jonathan P. How. Consensus-based auction approaches for decentralized task assignment. In *Proc. GNC*, Honolulu, HI, 2008.

[33] L. Busoniu, R. Babuska, and B. De Schutter. A comprehensive survey of multi-agent reinforcement learning. *IEEE Trans. SMC Part C*, 38(2):156–172, March 2008.

[34] Deng Cai, Xiaofei He, Ji-Rong Wen, and Wei-Ying Ma. Block-level link analysis. In *Proc. 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '04, pages 440–447. ACM, 2004.

[35] John Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (6):679–698, 1986.

[36] Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. Learning to rank: from pairwise approach to listwise approach. In *Proc. ICML*, pages 129–136. ACM, 2007.

[37] J. Carifio. Assigning students to career education programs by preference: scaling preference data for program assignments. 1, 1976.

[38] J. Carifio. Measuring vocational preferences: ranking versus categorical rating procedures. 3, 1976.

[39] Jennifer Casper and Robin Roberson Murphy. Human-robot interaction in rescue robotics. *IEEE Transaction on Systems, Man, and Cybernetics (SMCS)*, 34(2):138–153, 2004.

[40] Elkin Castro and Sanja Petrovic. Combined mathematical programming and heuristics for a radiotherapy pre-treatment scheduling problem. *Journal of Scheduling*, 15(3):333–346, 2012.

[41] Jessie YC Chen, Michael J Barnes, and Michelle Harper-Sciarini. Supervisory control of multiple robots: Human-performance issues and user-interface design. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 41(4):435–454, 2011.

[42] Jessie Y.C. Chen, Michael J. Barnes, and Zhihua Qu. Roboleader: an agent for supervisory control of mobile robots. In *Proc. International Conference on Human-Robot Interaction (HRI)*, 2010.

[43] Jiaqiong Chen and Ronald G. Askin. Project selection, scheduling and resource allocation with time dependent returns. *European Journal of Operational Research*, 193:23–34, 2009.

[44] J.Y.C. Chen, E.C. Haas, and M.J. Barnes. Human performance issues and user interface design for teleoperated robots. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 37(6):1231–1245, November 2007.

[45] Tsang-Hsiang Cheng, Chih-Ping Wei, and Vincent S. Tseng. Feature selection for medical data mining: Comparisons of expert judgment and automatic approaches. In *Proc. CBMS*, pages 165–170, 2006.

[46] Sonia Chernova and Manuela Veloso. Confidence-based policy learning from demonstration using gaussian mixture models. In *Proc. AAMAS*, pages 233:1–233:8. ACM, 2007.

[47] Sonia Chernova and Manuela Veloso. Multi-thresholded approach to demonstration selection for interactive robot learning. In *3rd ACM IEEE International Conference on Human-Robot Interaction (HRI)*. IEEE, March 2008.

[48] Steve Chien, Anthony Barrett, Tara Estlin, and Gregg Rabideau. A comparison of coordinated planning methods for cooperating rovers. In *Proc. Fourth International Conference on Autonomous Agents*, AGENTS '00, pages 100–101. ACM, 2000.

[49] Yoon Ho Cho, Jae Kyeong Kim, and Soung Hie Kim. A personalized recommender system based on web usage mining and decision tree induction. *Expert Systems with Applications*, 23(3):329 – 342, 2002.

[50] Uffe Gram Christensen and Anders Bjerg Pedersen. Lecture note on benders' decomposition. 2008.

[51] André Ciré, Elvin Coban, and John N Hooker. Mixed integer programming vs. logic-based benders decomposition for planning and scheduling. In *International Conference on AI and OR Techniques in Constriant Programming for Combinatorial Optimization Problems*, pages 325–331. Springer, 2013.

[52] Andrew S Clare, Mary L. Cummings, Jonathan P. How, Andrew K. Whitten, and Olivier Toupet. Operator objective function guidance for a real-time unmanned vehicle scheduling algorithm. *AIAA Journal of Aerospace Computing, Information and Communication*, 9:161–173, 2012.

[53] Mark Claypool, Anuja Gokhale, Tim Miranda, Pavel Murnikov, Dmitry Netes, and Matthew Sartin. Combining content-based and collaborative filters in an online newspaper. In *Proc. ACM SIGIR Workshop on Recommender Systems*, 1999.

[54] Jean-François Cordeau, Goran Stojković, François Soumis, and Jacques Desrosiers. Benders decomposition for simultaneous aicraft routing and crew scheduling. *Transportation Science*, 35(4):357–388, 2001.

[55] Mary L. Cummings, Amy S. Brzezinski, and John D. Lee. Operator performance and intelligent aiding in unmanned aerial vehicle scheduling. *IEEE Intelligent Systems*, 22(2):52–59, March 2007.

[56] Mary L Cummings and Stephanie Guerlain. Developing operator capacity estimates for supervisory control of autonomous vehicles. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 49(1):1–15, 2007.

[57] Tapas Das, Abhijit Gosavi, Sridhar Mahadevan, and N. Marchalleck. Solving semi-markov decision problems using average reward reinforcement learning. *Management Science*, 45:560–574, 1999.

[58] E. J. de Visser, F. Krueger, P. McKnight, S. Scheid, M. Smith, S. Chalk, and R. Parasuraman. The world is not enough: Trust in cognitive agents. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 56(1):263–267, 2012.

[59] Rina Dechter, Italy Meiri, and Judea Pearl. Temporal constraint networks. *AI*, 49(1):61–91, 1991.

[60] Munjal Desai, Poornima Kaniarasu, Mikhail Medvedev, Aaron Steinfeld, and Holly Yanco. Impact of robot failures and feedback on real-time trust. In *Proceedings of the 8th ACM/IEEE International Conference on Human-robot Interaction*, HRI '13, pages 251–258, Piscataway, NJ, USA, 2013. IEEE Press.

[61] Munjal Desai, Mikhail Medvedev, Marynel Vázquez, Sean McSheehy, Sofia Gadea-Omelchenko, Christian Bruggeman, Aaron Steinfeld, and Holly Yanco. Effects of changing reliability on trust of robot systems. In *Human-Robot Interaction (HRI), 2012 7th ACM/IEEE International Conference on*, pages 73–80. IEEE, 2012.

[62] M Bernardine Dias. *TraderBots: A New Paradigm for Robust and Efficient Multirobot Coordination in Dynamic Environments*. PhD thesis, Robotics Institute, Carnegie Mellon University, January 2004.

[63] Nicole DiGiose. Hospitals hiring robots, February 2013.

[64] R. K. Dismukes, B. A. Berman, and L. D. Loukopoulous. *The Limits of Expertise: Rethinking Pilot Error and the Causes of Airline Accidents*. Ashgate Publishing, 2007.

[65] Stephen R Dixon and Christopher D Wickens. Automation reliability in unmanned aerial vehicle control: A reliance-compliance model of automation dependence in high workload. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 48(3):474–486, 2006.

[66] Anca Dragan, Kenton Lee, and Siddhartha Srinivasa. Teleoperation with intelligent and customizable interfaces. *Journal of Human-Robot Interaction, Special Issue on Technical and Social Advances in HRI: An Invitational Issue of JHRI*, 1(3), January 2013.

[67] Anca Dragan and Siddhartha Srinivasa. A policy blending formalism for shared control. *International Journal of Robotics Research*, May 2013.

[68] Anca D. Dragan and Siddhartha S. Srinivasa. Assistive teleoperation for manipulation tasks. In *Proceedings of the Seventh Annual ACM/IEEE International Conference on Human-Robot Interaction*, pages 123–124, New York, NY, USA, 2012. ACM.

[69] Andreas Drexl, Rüdiger Nissen, James H. Patterson, and Frank Salewski. Progen/πx – an instance generator for resource-constrained project scheduling problems with partially renewable resources and further extensions. *European Journal of Operational Research*, 125(1):59 – 72, 2000.

[70] Jill L Drury, Laurel Riek, and Nathan Rackliffe. A decomposition of uav-related situation awareness. In *Proceedings of the ACM SIGCHI/SIGART conference on Human-robot interaction*, pages 88–94. ACM, 2006.

[71] Richard O Duda and Peter E Hart. Use of the hough transformation to detect lines and curves in pictures. *Communications of the ACM*, 15(1):11–15, 1972.

[72] Edmund H. Durfee, James C. Boerkoel Jr., and Jason Sleight. Using hybrid scheduling for the semi-autonomous formation of expert teams. *Future Generation Computer Systems*, July 2013.

[73] Mica R Endsley. Design and evaluation for situation awareness enhancement. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, volume 32, pages 97–101. SAGE Publications, 1988.

[74] Mica R. Endsley. Situation awareness global assessment technique (sagat). In *Aerospace and Electronics Conference, 1988. NAECON 1988., Proceedings of the IEEE 1988 National*, pages 789–795 vol.3, May 1988.

[75] Mica R. Endsley. Toward a theory of situation awareness in dynamic systems. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 37(1):32–64, 1995.

[76] Mica R Endsley and DJ Garland. Theoretical underpinnings of situation awareness: A critical review. *Situation awareness analysis and measurement*, pages 3–32, 2000.

[77] Micar R. Endsley and David B. Kaber. Level of automation effects on performance, situation awareness, and workload in dynamic control task. 42(3):462–492, 1999.

[78] MR Ensdley. Toward a theory of situation awareness. *Human Factors*, 37:32–64, 1995.

[79] Elliot E. Entin and Daniel Serfaty. Adaptive team coordination. *Human Factors*, 41:312–325, 1999.

[80] George Ferguson, James F Allen, Bradford W Miller, et al. Trains-95: Towards a mixed-initiative planning assistant. In *In Proc. AIPS*, pages 70–77, 1996.

[81] Terrence Fong and Charles Thorpe. Vehicle teleoperation interfaces. *Autonomous Robots*, 11(1):9–18, 2001.

[82] Dieter Fox. Adapting the sample size in particle filters through kld-sampling. *IJRR*, 22:985–1003, 2003.

[83] Martin L Fracker. A theory of situation assessment: Implications for measuring situation awareness. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, volume 32, pages 102–106. SAGE Publications, 1988.

[84] Luca Maria Gambardella, Éric Taillard, and Giovanni Agazzi. MACS-VRPTW: A multiple colony system for vehicle routing problems with time windows. In *New Ideas in Optimization*, pages 63–76. McGraw-Hill, 1999.

[85] Arthur M. Geoffrion. Generalized benders decomposition. *Journal of Optimization Theory and Applications*, 10(4):237–260, 1972.

[86] Gene V. Glass, Perey D. Peckham, and James R. Sanders. Consequences of failure to meet assumptions underlying the analyses of variance and covariance. 1972.

[87] Matthew Gombolay, Reymundo Gutierrez, Shanelle Clarke, Giancarlo Sturla, and Julie Shah. Decision-making authority, team efficiency and human worker satisfaction in mixed human-robot teams. *AuRo*, 39(3):293–312, 2015.

[88] Matthew Gombolay, Reymundo Gutierrez, Giancarlo Sturla, and Julie Shah. Decision-making authority, team efficiency and human worker satisfaction in mixed human-robot teams. In *Proc. Robots: Science and Systems*, Berkeley, California, July 12-16, 2014.

[89] Matthew Gombolay, Reed Jensen, Jessica Stigile, Sung-Hyun Son, and Julie Shah. Decision-making authority, team efficiency and human worker satisfaction in mixed human-robot teams. In *Proc. IJCAI*, New York City, NY, U.S.A., July 9-15 2016.

[90] Matthew Gombolay and Julie Shah. Schedulability analysis of task sets with upper- and lower-bound temporal constraints. *Journal of Aerospace Information Systems*, 11(12):821–841, 2015.

[91] Matthew Gombolay, Ronald Wilcox, and Julie Shah. Fast scheduling of multi-robot teams with temporospatial constrints. In *Proc. RSS*, Berlin, Germany, June 24-28 2013.

[92] Michael A. Goodrich, Bryan S. Morse, Cameron Engh, Joseph L. Cooper, and Julie A. Adams. Towards using UAVs in wilderness search and rescue: Lessons from field trials. *Interaction Studies, Special Issue on Robots in the Wild: Exploring Human-Robot Interaction in Naturalistic Environments*, 10(3):453–478, 2009.

[93] Melanie L. De Grano, D. J. Medeiros, and David Eitel. Accommodating individual preferences in nurse scheduling via auctions and optimization. *Healthcare Management Science*, 12:228–242, September 2009.

[94] Masahiro Hamasaki, Hideaki Takeda, Ikki Ohmukai, and Ryutaro Ichise. Scheduling support system for academic conferences based on interpersonal networks. In *Proc. ACM Hypertext*, 2004.

[95] Iiro Harjunkooski and Ignacio E. Grossman. Decomposition techniques for multistage scheduling problems using mixed-integer and constraint programming methods. *Computers & Chemical Engineering*, 26:1533–1552, 2002.

[96] Sandra G. Hart and Lowell E. Staveland. Development of nasa-tlx (task load index): Results of empirical and theoretical research. In Peter A. Hancock and Najmedin Meshkati, editors, *Human Mental Workload*, volume 52 of *Advances in Psychology*, pages 139 – 183. North-Holland, 1988.

[97] Taher H. Haveliwala. Topic-sensitive PageRank. In *Proc. WWW*, pages 517–526. ACM, 2002.

[98] Kelsey P Hawkins, Sunny Bansal, Nam N Vo, and Aaron F Bobick. Anticipating human actions for collaboration in the presence of task and sensor uncertainty. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 2215–2222. IEEE, 2014.

[99] Thomas Haynes, Sandip Sen, Neeraj Arora, and Rajani Nadella. An automated meeting scheduling system that utilizes user preferences. In *Proc. First International Conference on Autonomous Agents*, AGENTS '97, pages 308–315. ACM, 1997.

[100] Ralf Herbrich, Thore Graepel, and Klaus Obermayer. *Large Margin Rank Boundaries for Ordinal Regression*, chapter 7, pages 115–132. MIT Press, January 2000.

[101] Laura Herlant, Rachel Holladay , and Siddhartha Srinivasa. Assistive teleoperation of robot arms via automatic time-optimal mode switching. In *Human-Robot Interaction*, March 2016.

[102] Jonathan L. Herlocker, Joseph A. Konstan, Loren G. Terveen, and John T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems*, 22(1):5–53, January 2004.

[103] Guy Hoffman. Evaluating fluency in human-robot collaboration. In *International Conference on Human-Robot Interaction (HRI), Workshop on Human Robot Collaboration*, 2013.

[104] Guy Hoffman and Cynthia Breazeal. Effects of anticipatory action on human-robot teamwork efficiency, fluency, and perception of team. In *Proceedings of the ACM/IEEE international conference on Human-robot interaction*, pages 1–8. ACM, 2007.

[105] Guy Hoffman and Wendy Ju. Designing robots with movement in mind. *Journal of Human-Robot Interaction*, 3(1):89–122, 2014.

[106] John N. Hooker. *Logic-Based Benders Decomposition, in Logic-Based Methods for Optimization: Combining Optimization and Constraint Satisfaction*. John Wiley & Sons, Inc., 2000.

[107] John N. Hooker. Logic-based benders decomposition. *Mathematical Programming*, 96:33–60, 2003.

[108] John N. Hooker. A hybrid method for planning and scheduling. Technical report, Pittsburgh, Tepper School of Business, Carnegie Mellon University, 2004.

[109] Eli R. Hooten, Sean T.Hayes, and Julie A. Adams. A comparison of communicative modes for map-based tasking. In *IEEE International Conference on Systems, Man, and Cybernetics*, 2011.

[110] Adam O Horvath and Leslie S Greenberg. Development and validation of the working alliance inventory. *Journal of counseling psychology*, 36(2):223, 1989.

[111] John Hu, Aaron Edsinger, Yi-Je Lim, Nick Donaldson, Mario Solano, Aaron Solochek, and Ronald Marchessault. An advanced medical robotic system augmenting healthcare capabilities-robotic nursing assistant. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 6264–6269. IEEE, 2011.

[112] Chien-Ming Huang and Bilge Mutlu. Learning-based modeling of multimodal behaviors for humanlike robots. In *Proc. HRI*, pages 57–64, 2014.

[113] Chien-Ming Huang and Bilge Mutlu. Anticipatory robot control for efficient human-robot collaboration. In *2016 11th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 83–90. IEEE, 2016.

[114] Mohamed Ibnkahla. Applications of neural networks to digital communications – a survey. *Signal Processing*, 80(7):1185 – 1215, 2000.

[115] Auke Jan Ijspeert, Jun Nakanishi, and Stefan Schaal. Movement imitation with nonlinear dynamical systems in humanoid robots. In *Proc. ICRA*, volume 2, pages 1398–1403, 2002.

[116] Tetsunari Inamura, Masayuki Inaba, and Hirochika Inoue. Acquisition of probabilistic behavior decision model based on the interactive teaching method. In *Proc. International Conference on Advanced Robotics*, 1999.

[117] Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proc. Symposium on Theory of computing*, pages 604–613, 1998.

[118] Vipul Jain and Ignacio E. Grossmann. Algorithms for hybrid MILP/CP models for a class of optimization problems. *Journal on Computing*, 13(4):258–276, 2001.

[119] J.-Y. Jian, A. M. Bisantz, and C. G. Drury. Foundations for an empirically determined scale of trust in automated systems. *International Journal of Cognitive Ergonomics*, 4(1):53–71, 2000.

[120] Rong Jin, Hamed Valizadegan, and Hang Li. Ranking refinement and its application to information retrieval. In *Proc. Conference on WWW*, pages 397–406, 2008.

[121] E.Gil Jones, M.Bernardine Dias, and Anthony Stentz. Time-extended multi-robot coordination for domains with intra-path constraints. *AuRo*, 30(1):41–56, 2011.

[122] Henry L. Jones, Stephen M. Rock, Dennis Burns, and Steve Morris. Autonomous robots in SWAT applications: Research, design, and operations challenges. *Association for Unmanned Vehicle Systems International*, 2002.

[123] Wendy Ju and David Sirkin. Animate objects: How physical motion encourages public interaction. In *Persuasive Technology*, pages 40–51. Springer, 2010.

[124] David B Kaber and Mica R Endsley. Out-of-the-loop performance problems and the use of intermediate levels of automation for improved control system functioning and safety. *Process Safety Progress*, 16(3):126–131, 1997.

[125] David B. Kaber and Mica R. Endsley. The effects of level of automation and adaptive automation on human performance, situation awareness and workload in a dynamic control task. *Theoretical Issues in Ergonomics Science*, 5(2):113–153, 2004.

[126] David B. Kaber, Emrah Onal, and Mica R. Endsley. Design of automation for telerobots and the effect on performance, operator situation awareness, and subjective workload. *Human Factors and Ergonomics in Manufacturing & Service Industries*, 10(4):409–430, 2000.

[127] Denise B Kandel. Homophily, selection, and socialization in adolescent friendships. *American journal of Sociology*, pages 427–436, 1978.

[128] Steven J Kass, Daniel A Herschler, and Michael A Companion. Are they shooting at me?: An approach to training situational awareness. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, volume 34, pages 1352–1356. SAGE Publications, 1990.

[129] Shannon M. Kehle, Nancy Greer, Indulis Rutks, and Timothy Wilt. Interventions to improve veterans? access to care: A systematic review of the literature. *Journal of General Internal Medicine*, 26(2):689–696, 2011.

[130] Cory D Kidd and Cynthia Breazeal. Effect of a robot on user perceptions. In *Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, volume 4, pages 3559–3564. IEEE, 2004.

[131] Sara Kiesler, Aaron Powers, Susan R Fussell, and Cristen Torrey. Anthropomorphic interactions with a robot and robot-like agent. *Social Cognition*, 26(2):169–181, 2008.

[132] Eunji Kim, Jonathan Sangyun Lee, Sukjae Choi, and Ohbyung Kwon. Human compliance with task-oriented dialog in social robot interaction. In *Proceedings of the Tenth Annual ACM/IEEE International Conference on Human-Robot Interaction Extended Abstracts*, pages 3–4. ACM, 2015.

[133] Hyea Kyeong Kim, Jae Kyeong Kim, and Y.U. Ryu. Personalized recommendation over a customer network for ubiquitous shopping. *IEEE Transactions on Services Computing*, 2, April 2009.

[134] Jae Kyeong Kim, Yoon Ho Cho, Woo Ju Kim, Je Ran Kim, and Ji Hae Suh. A personalized recommendation procedure for internet shopping support. *Electronic Commerce Research and Applications*, 1(3?4):301 – 313, 2002.

[135] Scott R Klemmer, Björn Hartmann, and Leila Takayama. How bodies matter: five themes for interaction design. In *Proceedings of the 6th conference on Designing Interactive systems*, pages 140–149. ACM, 2006.

[136] George Konidaris and Andrew Barto. Building portable options: Skill transfer in reinforcement learning. In *Proc. IJCAI*, pages 895–900, 2007.

[137] George Konidaris, Sarah Osentoski, and Philip Thomas. Value function approximation in reinforcement learning using the fourier basis. In *Proc. AAAI*, pages 380–385, 2011.

[138] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, August 2009.

[139] G. Ayorkor Korsah, Anthony Stentz, and M. Bernardine Dias. A comprehensive taxonomy for multi-robot task allocation. *IJRR*, 32(12):1495–1512, 2013.

[140] Arthur F Kramer. Physiological metrics of mental workload: A review of recent progress. *Multiple-task performance*, pages 279–328, 1991.

[141] Alex Kushleyev, Daniel Mellinger, Caitlin Powers, and Vijay Kumar. Towards a swarm of agile micro quadrotors. *Autonomous Robots*, 35(4):287–300, 2013.

[142] Woo Young Kwon and Il Hong Suh. A temporal bayesian network with application to design of a proactive robotic assistant. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 3685–3690. IEEE, 2012.

[143] Niels Landwehr, Mark Hall, and Eibe Frank. Logistic model trees. *Machine Learning*, 59(1-2):161–205, 2005.

[144] Przemyslaw A. Lasota and Julie A. Shah. Analyzing the effects of human-aware motion planning on close-proximity human–robot collaboration. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 57(1):21–33, 2015.

[145] John D. Lee and Katrina A. See. Trust in automation: Designing for appropriate reliance. *Human Factors*, 46(1):50–80, 2004.

[146] John D Lee and Katrina A See. Trust in automation: Designing for appropriate reliance. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 46(1):50–80, 2004.

[147] K. W. Lee, W. Peng, S.-A. Jin, and C. Yan. Can robots manifest personality?: An empirical test of personality recognition, social responses, and social presence in humanrobot interaction. *Journal of Communication*, 56(4):754–772, 2006.

[148] Kwan Min Lee, Younbo Jung, Jaywoo Kim, and Sang Ryong Kim. Are physically embodied social agents better than disembodied social agents?: The effects of physical embodiment, tactile interaction, and people's loneliness in human-robot interaction. *International Journal of Human-Computer Studies*, 64(10):962–973, 2006.

[149] Kwan Min Lee, Wei Peng, Seung-A Jin, and Chang Yan. Can robots manifest personality?: An empirical test of personality recognition, social responses, and social presence in humanrobot interaction. *Journal of communication*, 56(4):754–772, 2006.

[150] Zne-Jung Lee, Shun-Feng Su, and Chou-Yuan Lee. Efficiently solving general weapon-target assignment problem by genetic algorithms with greedy eugenics. *IEEE Trans. SMC Part B*, 33(1):113–121, Feb 2003.

[151] Thomas Lemaire, Rachid Alami, and Simon Lacroix. A distributed tasks allocation scheme in multi-uav context. In *Robotics and Automation, 2004. Proc.. ICRA '04. 2004 IEEE International Conference on*, volume 4, pages 3622–3627 Vol.4, April 2004.

[152] Jan Karel Lenstra and Alexander H. G. Rinnooy Kan. Complexity of scheduling under precedence constraints. *Operations Research*, 26(1):22–35, 1978.

[153] Vladimir I Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pages 707–710, 1966.

[154] Daniel Leyzberg, Samuel Spaulding, and Brian Scassellati. Personalizing robot tutors to individuals' learning differences. In *Proceedings of the 2014 ACM/IEEE international conference on Human-robot interaction*, pages 423–430. ACM, 2014.

[155] Haitao Li and Keith Womer. Scheduling projects with multi-skill personnel by a hybrid milp/cp benders decomposition algorithm. *Journal of Scheduling*, 12:281–298, 2008.

[156] Ping Li, Qiang Wu, and Christopher J. Burges. Mcrank: Learning to rank using multiple classification and gradient boosting. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *NIPS*, pages 897–904. MIT Press, Cambridge, MA, 2007.

[157] Wu Lihua, Liu Lu, Li Jing, and Li Zongyong. Modeling user multiple interests by an improved {GCS} approach. *Expert Systems with Applications*, 29(4):757–767, 2005.

[158] Gordon S. Linoff and Michael J.A. Berry. *Data Mining Techniques: For Marketing, Sales and Customer Relationship Management*. Wiley, Hoboken, New Jersey, 2004.

[159] Chang Liu, Jessica B Hamrick, Jaime F Fisac, Anca D Dragan, J Karl Hedrick, S Shankar Sastry, and Thomas L Griffiths. Goal inference improves objective and perceived performance in human-robot collaboration. In *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*, pages 940–948. International Foundation for Autonomous Agents and Multiagent Systems, 2016.

[160] Lantao Liu and Dylan A. Shell. Optiml market-based multi-robot task allocation via strategic pricing. In *Proc. RSS*, Berlin, Germany, June 24-28 2013.

[161] Shayne Loft, Penelope Sanderson, Andrew Neal, and Martijn Mooij. Modeling and predicting mental workload in en route air traffic control: Critical review and broader implications. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 49(3):376–399, 2007.

[162] Kiran Lokhande and Hayley J. Davison Reynolds. Cognitive workload and visual attention analyses of the air traffic control tower flight data manager (tfdm) prototype demonstration. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 56(1):105–109, 2012.

[163] C. Lottaz. Constraint solving, preference activation and solution adaptation in idiom. Technical report, 1996.

[164] Santiago Macho, Marc Torrens, and Boi Faltings. A multi-agent recommender system for planning meetings. In *Proc. ACM Conference on Autonomous Agents, Workshop on Agent-based Recommender Systems*, 2000.

[165] Colin F. Mackenzie, Yan Xiao, and Richard Horst. Video task analysis in high performance teams. *Cognition, Technology, and Work*, 6:139–147, 2004.

[166] Maxim Makatchev, Reid Simmons, Majd Sakr, and Micheline Ziadee. Expressing ethnicity through behaviors of a robot character. In *Proceedings of the 8th*

ACM/IEEE international conference on Human-robot interaction, pages 357–364. IEEE Press, 2013.

[167] Naresh K Malhotra. *Marketing Research: an Applied Orientation*. Prentice Hall, Upper Saddle River, New Jersey, 2010.

[168] Richard Kipp Martin. *Large Scale Linear and Integer Optimization: A unified Approach*. Kluwer Acadmic Publishers, Dordrecht, the Netherlands, 1999.

[169] Mitchell McIntire, Ernesto Nunes, and Maria Gini. Iterated multi-robot auctions for precedence-constrained task scheduling. In *Proc. AAMAS*, pages 1078–1086, 2016.

[170] Miller McPherson, Lynn Smith-Lovin, and James M Cook. Birds of a feather: Homophily in social networks. *Annual review of sociology*, pages 415–444, 2001.

[171] Anne Mercier, Jean-Franois Cordeau, and Franois Soumis. A computational study of benders decomposition for the integrated aircraft routing and crew scheduling problem. *Computers & Operations Research*, 32(6):1451 – 1476, 2005.

[172] Bernard Michini and Jonathan P. How. Bayesian nonparametric inverse reinforcement learning. In *Machine Learning and Knowledge Discovery in Databases*, volume 7524 of *Lecture Notes in Computer Science*, pages 148–163. Springer Berlin Heidelberg, 2012.

[173] Benny Morris. Should israel and the u.s. rethink iron dome's usefulness? *Los Angeles Times*, August 21.

[174] Katharina Muelling, Arun Venkatraman, Jean-Sebastien Valois, John Downey, Jeffrey Weiss, Shervin Javdani, Martial Hebert, Andrew B Schwartz, Jennifer L Collinger, and J Andrew Bagnell. Autonomy infused teleoperation with application to bci manipulation. In *Proceedings of Robotics: Science and Systems*, 2015.

[175] Ryota Murai, Tadashi Sakai, Yuma Honda, et al. Recognition of 3d dynamic environments for mobile robot by selective memory intake and release of data from 2d sensors. In *System Integration (SII), 2012 IEEE/SICE International Symposium on*, pages 621–628. IEEE, 2012.

[176] Ryota Murai, Tadashi Sakai, Hiroyuki Kawano, Yoshihiko Matsukawa, Yuma Honda, KC Campbell, et al. A novel visible light communication system for enhanced control of autonomous delivery robots in a hospital. In *System Integration (SII), 2012 IEEE/SICE International Symposium on*, pages 510–516. IEEE, 2012.

[177] Nicola Muscettola, Paul Morris, and Ioannis Tsamardinos. Reformulating temporal plans for efficient execution. In *Proc. KR&R*, Trento, Italy, June 2-5, 1998.

[178] Bilge Mutlu and Jodi Forlizzi. Robots in organizations: the role of workflow, social, and environmental factors in human-robot interaction. In *Human-Robot Interaction (HRI), 2008 3rd ACM/IEEE International Conference on*, pages 287–294. IEEE, 2008.

[179] National Transportation Safety Board. Aircraft accident report: Eastern Airlines 401/L-1011, Miami, Florida, December 29, 1972. Number NTSB/AAR-73-14. Washington, DC, 1973.

[180] National Transportation Safety Board. Aircraft accident report: United Airlines, inc., McDonnell Douglass DC-8-61, N8082U, Portland, Oregon, December 28, 1978. Number NTSB/AAR-79-07. Washington, DC, 1979.

[181] National Transportation Safety Board. Aircraft separation incidents at Hartsfield Atlanta International Airport, Atlanta, Georgia. Number NTSB/SIR-81-6. Washington, DC, 1981.

[182] National Transportation Safety Board. Aircraft accident report: Northwest airlines, inc. McDonnell-Douglous dc-9-82, N312RC, Detroit Metropolitan Wayne County Airport, August 16, 1987. Number NTSB/AAR-99-05. Washington, DC, 1988.

[183] National Transportation Safety Board. Aircraft accident report: US Air Flight 105, Boeing 737-200, N282AU, Kansas International Airport, Missouri, September 8, 1989. Number NTSB/AAR-90-04. Washington, DC, 1990.

[184] Anh Nguyen, Jason Yosinski, and Jeff Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 427–436. IEEE, 2015.

[185] Ute Niederée, Meike Jipp, Uwe Teegen, and Mark Vollrath. Effects of observability, mood states, and workload on human handling errors when monitoring aircraft automation. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 56(1):1481–1485, 2012.

[186] Stefanos Nikolaidis, Keren Gu, Ramya Ramakrishnan, and Julie Shah. Efficient model learning from joint-action demonstration for human-robot collaborative tasks. In *Proc. International Conference on Human-Robot Interaction (HRI)*, 2015.

[187] Stefanos Nikolaidis and Julie Shah. Human-robot cross-training: computational formulation, modeling and evaluation of a human team training strategy. In *Proc. International Conference on Human-Robot Interaction (HRI)*, pages 33–40, 2013.

[188] Ernesto Nunes and Maria Gini. Multi-robot auctions for allocation of tasks with temporal constraints. In *Proc. AAAI*, pages 2110–2116, 2015.

[189] P. Odom and S. Natarajan. Active advice seeking for inverse reinforcement learning. In *Proc. AAAI*, pages 4186–4187, 2015.

[190] Dan R Olsen Jr and Stephen Bart Wood. Fan-out: measuring human control of multiple robots. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 231–238. ACM, 2004.

[191] Ali Gürcan Özkil, Zhun Fan, Steen Dawids, H Aanes, Jens Klæstrup Kristensen, and Kim Hardam Christensen. Service robots for hospitals: A case study of transportation tasks in a hospital. In *Automation and Logistics, 2009. ICAL'09. IEEE International Conference on*, pages 289–294. IEEE, 2009.

[192] Meltem Öztürké, Alexis Tsoukiìs, and Philippe Vincke. Preference modelling. In *Multiple Criteria Decision Analysis: State of the Art Surveys*, volume 78 of *International Series in Operations Research & Management Science*, pages 27–59. Springer New York, 2005.

[193] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The PageRank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab, November 1999. SIDL-WP-1999-0120.

[194] Tapio Pahikkala, Evgeni Tsivtsivadze, Antti Airola, Jorma Boberg, and Tapio Salakoski. Learning to rank with pairwise regularized least-squares. In *SIGIR Workshop on Learning to Rank for Information Retrieval*, pages 27–33, 2007.

[195] R. Pak, N. Fink, M. Price, B. Bass, and L. Sturre. Decision support aids with anthropomorphic characteristics influence trust and performance in younger and older adults. *Ergonomics*, 55(9):1059–1072, 2012.

[196] R. Parasuraman, T.B. Sheridan, and Christopher D. Wickens. A model for types and levels of human interaction with automation. *Trans. SMC-A*, 30(3):286–297, 2000.

[197] Raja Parasuraman and Victor Riley. Humans and automation: Use, misuse, disuse, abuse. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 39(2):230–253, 1997.

[198] Raja Parasuraman, Thomas B. Sheridan, and Christopher D. Wickens. Situation awareness, mental workload, and trust in automation: Viable, empirically supported cognitive engineering constructs. *Journal of Cognitive Engineering and Decision Making*, 2(2):140–160, 2008.

[199] Deuk Hee Park, Hyea Kyeong Kim, Il Young Choi, and Jae Kyeong Kim. A literature review and classification of recommender systems research. *Expert Systems with Applications*, 39(11):10059 – 10072, 2012.

[200] James Parker, Ernesto Nunes, Julio Godoy, and Maria Gini. Exploiting spatial locality and heterogeneity of agents for search and rescue teamwork. *Journal of Field Robotics*, 2015.

[201] James H. Patterson. A comparison of exact approaches for solving the multiple constrained resource, project scheduling problem. *Management Science*, 30(7):854 – 867, 1984.

[202] R.M. Pierce and K.J. Kuchenbecker. A data-driven method for determining natural human-robot motion mappings in teleoperation. In *Biomedical Robotics and Biomechatronics (BioRob), 2012 4th IEEE RAS EMBS International Conference on*, pages 169–176, June 2012.

[203] Steven D. Pizer and Julia C. Prentice. What are the consequences of waiting for health care in the veteran population? *Journal of General Internal Medicine*, 26(2):676–682, 2011.

[204] Robert W. Proctor and Trisha Van Zandt. *Human Factors in Simple and Complex Systems*. CRC Press, Boca Raton, FL, 2008.

[205] Martin L Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

[206] Hema Raghavan, Omid Madani, and Rosie Jones. Active learning with feedback on features and instances. *Journal of Machine Learning Research*, 7:1655–1686, December 2006.

[207] Deepak Ramachandran and Eyal Amir. Bayesian inverse reinforcement learning. In *Proc. IJCAI*, pages 2586–2591, 2007.

[208] Varun Ramanujam and Hamsa Balakrishnan. Estimation of maximum-likelihood discrete-choice models of the runway configuration selection process. In *Proc. ACC*, pages 2160–2167, June 2011.

[209] Monia Rekik, Jean-François Cordeau, and François Soumis. Consensus-based decentralized auctions for robust task allocation. *IEEE Transactions on Robotics*, 25:912–926, 2004.

[210] Monia Rekik, Jean-François Cordeau, and François Soumis. Using benders decomposition to implicitly model tour scheduling. *Transportation Science*, 128:111–133, 2004.

[211] Huizhi Ren and Lixin Tang. An improved hybrid MILP/CP algorithm framework for the job-shop scheduling. In *Proc. IEEE International Conference on Automation and Logistics (ICAL)*, Shenyang, 2009.

[212] Paul Reverdy, Vaibhav Srivastava, and Naomi Ehrich Leonard. Modeling human decision-making in generalized gaussian multi-armed bandits. *CoRR*, abs/1307.6134, 2013.

[213] Laurel D Riek and Peter Robinson. Challenges and opportunities in building socially intelligent machines. *IEEE Signal Processing Magazine*, 28(3):146–149, 2011.

[214] Jennifer M. Riley, David B. Kaber, and John V. Draper. Situation awareness and attention allocation measures for quantifying telepresence experiences in teleoperation. *Human Factors and Ergonomics in Manufacturing & Service Industries*, 14(1):51–67, 2004.

[215] P. Robinette, A. M. Howard, and A. R. Wagner. Overtrust of robots in emergency evacuation scenarios. In *Proc. HRI*, 2016.

[216] Everett M Rogers and Dilip K Bhowmik. Homophily-heterophily: Relational concepts for communication research. *Public opinion quarterly*, 34(4):523–538, 1970.

[217] Jason C. Ryan, Ashis Gopal Banerjee, Mary L. Cummings, and Nicholas Roy. Comparing the performance of expert user heuristics and an integer linear program in aircraft carrier deck operations. *IEEE Transaction on Cybernetics*, (9), August 2013.

[218] Paul E. Rybski and Richard M. Voyles. Interactive task training of a mobile robot through human gesture recognition. In *IEEE International Conference on Robotics and Automation*, pages 664–669, 1999.

[219] Eduardo Salas, Jennifer E. Fowlkes, Renee J. Stout, Dana M. Milanovich, and Carolyn Prince. Does CRM training improve teamwork skills in the cockpit?: Two evaluation studies. *Human Factors*, 41:326–343, 1999.

[220] Claude Sammut, Scott Hurst, Dana Kedzier, and Donal Michie. Learning to fly. In *Proc. ICML*, pages 385–393, July 1992.

[221] Penelope M Sanderson. The human planning and scheduling role in advanced manufacturing systems: an emerging human factors domain. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 31(6):635–666, 1989.

[222] Nadine B Sarter and David D Woods. Situation awareness: A critical but ill-defined phenomenon. *The International Journal of Aviation Psychology*, 1(1):45–57, 1991.

[223] Badrul M. Sarwar, George Karypis, Joseph A. Konstan, and John T. Riedl. Application of dimensionality reduction in recommender system - a case study. In *Proc. ACM WEBKDD workshop*, 2000.

[224] Joe Saunders, Chrystopher L. Nehaniv, and Kerstin Dautenhahn. Teaching robots by moulding behavior and scaffolding the environment. In *Proc. 1st ACM SIGCHI/SIGART Conference on Human-Robot Interaction*, pages 118–125. ACM, 2006.

[225] J. Ben Schafer, Dan Frankowski, Jon Herlocker, and Shilad Sen. Collaborative filtering recommender systems. In *The Adaptive Web*, pages 291–324. Springer-Verlag, Berlin, Heidelberg, 2007.

[226] Marc Schröder and Jürgen Trouvain. The german text-to-speech synthesis system mary: A tool for research, development and teaching. *International Journal of Speech Technology*, 6(4):365–377, 2003.

[227] Smruti J. Shah, James P. Bliss, Eric T. Chancey, and J. Christopher Brill. Effects of alarm modality and alarm reliability on workload, trust, and driving performance. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 59(1):1535–1539, 2015.

[228] Thomas B Sheridan. Adaptive automation, level of automation, allocation authority, supervisory control, and adaptive control: Distinctions and modes of adaptation. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 41(4):662–667, 2011.

[229] Scott A. Shipman and Christine A. Sinsky. Expanding primary care capacity by reducing waste and improving efficiency of care. *Health Affairs (Millwood)*, 32(11):1990–1997, 2013.

[230] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.

[231] Marius M. Solomon. Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research*, 35(2):254–265, 1987.

[232] M.J. Soomer and G.J. Franx. Scheduling aircraft landing using airlines' preferences. *European Journal of Operational Research*, 190:277–291, October 2008.

[233] Neville A. Stanton and Mark S. Young. Automatic intelligent cruise control. journal of intelligent systems. *Journal of Intelligent Systems*, 15:357–388, 2011.

[234] Aaron Steinfeld, Terrence Fong, David Kaber, Michael Lewis, Jean Scholtz, Alan Schultz, and Michael Goodrich. Common metrics for human-robot interaction. In *Proceedings of the 1st ACM SIGCHI/SIGART Conference on Human-robot Interaction*, HRI '06, pages 33–40, New York, NY, USA, 2006. ACM.

[235] Cynthia Sung, Nora Ayanian, and Daniela Rus. Improving the performance of multi-robot systems by task switching. In *Proc. ICRA*, Karlsruhe, Germany, May 6-10 2013.

[236] Richard S Sutton, David A McAllester, Satinder P Singh, Yishay Mansour, et al. Policy gradient methods for reinforcement learning with function approximation. In *Proc. NIPS*, pages 1057–1063, 1999.

[237] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[238] Peter Szolovits, Ramesh S. Patil, and William B. Schwartz. Artificial intelligence in medical diagnosis. *Annals of Internal Medicine*, 108(1):80–87, 1988.

[239] Peter Szolovits and Stephen G. Pauker. Categorical and probabilistic reasoning in medical diagnosis. *Artificial Intelligence*, 11(1–2):115–144, 1978. Applications to the Sciences and Medicine.

[240] Leila Takayama and Caroline Pantofaru. Influences on proxemic behaviors in human-robot interaction. In *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, pages 5495–5502. IEEE, 2009.

[241] K.C. Tan, L.H. Lee, Q.L. Zhu, and K. Ou. Heuristic methods for vehicle routing problem with time windows. *Artificial Intelligence in Engineering*, 15(3):281 – 295, 2001.

[242] Wei Tan. Integration of process planning and scheduling - a review. *Journal of Intelligent Manufacturing*, 11:51–63, 2000.

[243] Wei Tan. A linearized polynomial mixed integer programming model for the integration of process planning and scheduling. *Journal of Intelligent Manufacturing*, 15:593–605, 2004.

[244] Fang Tang and Lynne E. Parker. ASyMTRe: Automated synthesis of multi-robot task solutions through software reconfiguration. In *Proc. ICRA*, May 2005.

[245] Adriana Tapus, Cristian Tapus, and Maja Matarić. The role of physical embodiment of a therapist robot for individuals with cognitive impairments. In *Robot and Human Interactive Communication, 2009. RO-MAN 2009. The 18th IEEE International Symposium on*, pages 103–107. IEEE, 2009.

[246] Stefanie Tellex, Ross A. Knepper, Adrian Li, Nicholas Roy, and Daniela Rus. Asking for help using inverse semantics. In *Proceedings of the Robotics Science and Systems (RSS) Conference*, Berkeley, CA, July 2014.

[247] Allison Terrell and Bilge Mutlu. A regression-based approach to modeling addressee backchannels. In *Proc. Special Interest Group on Discourse and Dialogue*, pages 280–289, 2012.

[248] Andrea L. Thomaz and Cynthia Breazeal. Reinforcement learning with human teachers: Evidence of feedback and guidance with implications for learning performance. In *Proc. AAAI*, pages 1000–1005, 2006.

[249] Pamela S. Tsang and Michael A. Vidulich. *Mental Workload and Situation Awareness*, pages 243–268. John Wiley & Sons, Inc., 2006.

[250] Vaibhav V Unhelkar and Julia A Shah. Contact: Deciding to communicate during time-critical collaborative tasks in unknown, deterministic domains. In *In Proc. Association for the Advancement of Artificial Intelligence*, 2016.

[251] Hamed Valizadegan, Rong Jin, Ruofei Zhang, and Jianchang Mao. Learning to rank by optimizing NDCG measure. In *NIPS*, pages 1883–1891, 2009.

[252] Adam Vogel, Deepak Ramach, Rakesh Gupta, and Antoine Raux. Improving hybrid vehicle fuel efficiency using inverse reinforcement learning. In *Proc. AAAI*, pages 384–390, 2012.

[253] Maksims N. Volkovs and Richard S. Zemel. Boltzrank: Learning to maximize expected ranking gain. In *Proc. ICML*, pages 1089–1096, 2009.

[254] C. Volpe, J. Cannon-Bowers, E. Salas, and P. Spector. The impact of cross-training on team functioning: an empirical investigation. *Human Factors*, 38:87–100, 1996.

[255] Yi-Chi Wang and John M. Usher. Application of reinforcement learning for agent-based production scheduling. *Eng. Appl. Artif. Intell.*, 18(1):73–82, February 2005.

[256] C. D. Wickens, J. G. Hollands, S. Banbury, and R. Parasuraman. *Engineering psychology and human performance*. Pearson Education, 2013.

[257] Christopher D. Wickens. Multiple resources and mental workload. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 50(3):449–455, 2008.

[258] Christopher D Wickens, Huiyang Li, Amy Santamaria, Angelia Sebok, and Nadine B Sarter. Stages and levels of automation: An integrated meta-analysis. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, volume 54, pages 389–393. SAGE Publications, 2010.

[259] Ronald J. Wilcox, Stefanos Nikolaidis, and Julie A. Shah. Optimization of temporal dynamics for adaptive human-robot interaction in assembly manufacturing. In *Proc. RSS*, Sydney, Australia, July 9-13 2012.

[260] Ronald J. Wilcox and Julie A. Shah. Optimization of multi-agent workflow for human-robot collaboration in assembly manufacturing. In *Proc. AIAA Infotech@Aerospace*, 2012.

[261] Kevin D Williams. The effects of homophily, identification, and violent video games on players. *Mass Communication and Society*, 14(1):3–24, 2010.

[262] Jun Wu, Xin Xu, Pengcheng Zhang, and Chunming Liu. A novel multi-agent reinforcement learning approach for job scheduling in grid computing. *Future Generation Computer Systems*, 27(5):430 – 439, 2011.

[263] Shun-Zheng Yu. Hidden semi-markov models. *Artificial Intelligence*, 174(2):215 – 243, 2010. Special Review Issue.

[264] M. M. Zavlanos, L. Spesivtsev, and G. J. Pappas. A distributed auction algorithm for the assignment problem. In *Proc. 47^{th} IEEE Conference on Decision and Control*, Cancun, Mexico, 2008.

[265] Chongjie Zhang and Julie Shah. On fairness in decision-making under uncertainty: Definitions, computation, and comparison, 2015.

[266] F. Zhang. Schedulability analysis for real-time systems with edf scheduling. 58:1250–1258, September 2009.

[267] Haoqi Zhang, Edith Law, Rob Miller, Krzysztof Gajos, David Parkes, and Eric Horvitz. Human computation tasks with global constraints. In *Proc. SIGCHI Conference on Human Factors in Computing Systems*, CHI '12, pages 217–226. ACM, 2012.

[268] Wei Zhang and Thomas G. Dietterich. A reinforcement learning approach to job-shop scheduling. In *Proc. IJCAI*, pages 1114–1120, 1995.

[269] Jiangchuan Zheng, Siyuan Liu, and Lionel Ni. Robust bayesian inverse reinforcement learning with sparse behavior noise. In *Proc. AAAI*, pages 2198–2205, 2015.

[270] Brian D. Ziebart, Andrew Maas, J. Andrew Bagnell, and Anind K. Dey. Maximum entropy inverse reinforcement learning. In *Proc. AAAI*, pages 1433–1438, 2008.