Integrated Mission Specification and Task Allocation for Robot Teams - Part 2: Testing and Evaluation

Patrick Ulam, Yochiro Endo, Alan Wagner, Ronald Arkin
College of Computing
Georgia Institute of Technology
Atlanta, USA
Email: {pulam, endo, alan.wagner, arkin}@cc.gatech.edu

***Abstract*— This work presents the evaluation of two mission specification and task allocation architectures. These architectures, described in part 1 of this paper, present novel means with which to integrate a case-based reasoning (CBR) mission planner with contract net protocol (CNP) based task allocation. In the first design, the CBR and runtime-CNP architecture, the case-based mission planner generates mission plans that support necessary behaviors for CNP-based task allocation and execution. In the second design, the CBR and premission-CNP architecture, task allocation takes place during mission specification. The results of an empirical evaluation of the CBR and runtime-CNP across three naval scenarios is described. Finally, we briefly describe an earlier usability evaluation of the CBR and premission-CNP architecture using goals, operators, methods, and selection rules modeling.**

I. Introduction

Fielding teams of highly heterogeneous robots poses many logistical challenges for their operators. As these teams and their missions grow more complex, it becomes increasingly important to examine systems in which this complexity can be offloaded from the user. This work evaluates two such systems. In the first, a case-based planner is used to assist the user in generating complex multi-robot missions that support runtime task allocation via CNP-based negotiation [1]. This system, the CBR and runtime-CNP architecture, allows for the rapid creation of missions where task assignment is deferred until needed, thus allowing the team to better handle dynamic environments with uncertain objectives. The second system, the CBR and premission-CNP architecture, integrates task allocation and mission specification during the mission creation process. Leveraging the user's knowledge of the component tasks of the mission, the CBR and premission-CNP architecture creates mappings between the tasks and robots that ensure the mission goals are achievable by the team.

Part 2 of this work presents the empirical evaluation of both the CBR and runtime-CNP architecture and the CBR and premission-CNP architecture while a detailed description of both architectures is provided earlier in part 1 [2]. Here, the performance of the CBR and runtime-CNP system is examined in the context of three naval scenarios: naval mine clearance, vessel interception, and target tracking. Finally, an earlier usability evaluation of the CBR and premission-CNP system is discussed.

II. Background and Related Work

Mission specification, the process in which a detailed series of steps to accomplish a mission is generated, can be an arduous process for those who field multi-robot systems. As a result, a number of mission specification systems and tools have been developed to aid the user in generating these plans. One such tool, used as the basis for this work, is the MissionLab robotics software toolset [3].

MissionLab presents a graphical interface with which a user can generate FSA mission plans for one or more robots. Recent research has looked at incorporating case-based reasoning tools within MissionLab to further streamline the mission specification process [4]. The architectures evaluated here, and detailed in part 1 of this paper, expand on this work. These CBR/CNP architectures augment the case-based planner with market-based task allocation to assist in creation of missions for highly heterogeneous teams in uncertain environments.

Both the premission and runtime CBR/CNP architectures evaluated afford different advantages to the user. The premission architecture, for instance, allows the user to rapidly specify complex multi-task, multi-robot missions via a task-based interface. While many systems exist which combine off-line mission planning and task allocation [5][6], the premission architecture differs from them in several regards. Many of these architectures integrate task allocation directly in to the planning process via constraint matching. The CBR and premission-CNP system, on the otherhand, is capable of making finer-grained allocation decisions by not only determining if a robot can perform the task, but also how well it can perform the task. A notable exception to this, the M+ architecture [7], provides similar task decomposition and allocation before mission execution. The CBR and premission-CNP architecture, however, differentiates itself through the use of a powerful interface with which the user can easily generate, modify, verify, and reuse mission specifications.

The runtime architecture's strength, on the other hand, lies in the ability to generate missions when the user may not know the details of the mission objectives during specification. The CBR and runtime-CNP architecture attempts to address this problem by aiding the user in the specification of missions that encapsulate on-line market-based task allocation [8]. A number of market-based task allocation architectures have been created, many of which are discussed in [8] and [9]. Of these, however, the most closely related are M+ [7] and MURDOCH [10]. The M+ architecture provides both task decomposition as well as on-line task reallocation in a manner similar to the runtime system. The M+ architecture assumes, however, that the tasks and a partial task ordering are known at the time of mission specification. The CBR and runtime-CNP system task allocation strategy, similar to that used in MURDOCH, generate the robot-to-task mappings at runtime as the tasks are introduced into the environment. The runtime architecture and MURDOCH do differ, however, in their task reallocation strategy.

III. Evaluation

Because task allocation is integrated with mission specification at different timeframes for each of the two designs (during mission execution vs. during mission specification), the evaluation of each necessitates a different approach. The CBR and runtime-CNP design integrates task allocation during mission execution. We evaluate this approach through a series of experiments conducted in simulation using the MissionLab robotic software toolset. These experiments evaluate the CBR and runtime-CNP architecture against two different baseline systems in which the task allocation system is altered. The CBR and premission-CNP design provides task allocation during the mission specification stage. Therefore, we discuss the usability impact of this design on the mission specification process.

A. CBR and Runtime-CNP Design Evaluation

In order to evaluate the integrated CBR and runtime-CNP architecture four sets of experiments were conducted. The first two experiments compare the CBR and runtime-CNP system's task allocation strategy against that of a baseline system in a naval mine clearance (NMC) and target interception domain. The third and fourth experiments compare the CBR and runtime-CNP architecture against a similar system in which the ability to reallocate tasks in response to dynamic mission parameters is lesioned. The experimental setup of each of these is described below.

1) Random Allocation Experiments: The first two experiments conducted with the CBR and runtime-CNP system test our hypothesis that the intelligent allocation decisions of CBR and runtime-CNP architecture will increase mission performance over alternative allocation strategies utilizing less information. Additionally, we hypothesize that any advantages conferred by the CBR and runtime-CNP architecture will increase with the team size when compared to the baseline system. The domains chosen to test these hypotheses were naval mine clearance and target interception.

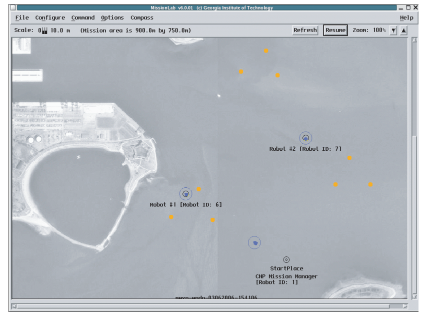


Fig. 1. Overview of the mission area for the NMC scenarios. Circles indicate the initial deployment points of the robots.

In the NMC mission, the team of robots locates nine mines within the mission area (figure 1). The team is tasked with approaching the mines and then neutralizing them. Team sizes of two to six robots were used in addition to a command and control vehicle which monitors the mission progress. During mission execution, the command and control vehicle is responsible for reporting the location of each mine found. Upon discovery of a mine, the command and control vehicle generates a call for proposals detailing the position of the mine. The available robots in the mission area place bids on the mine disposal task where the bid for each robot n is calculated as $\frac{1}{dist(robot_n, mine)}$. The robot with the highest bid wins the contract to disarm that particular mine. This process is continued until all the mines in the mission area have been neutralized. The experiment consisted of 50 trials. Each set of fifty trials was repeated with teams ranging from 2 to 6 robots placed in random valid positions (i.e. unmanned surface vehicles are not placed on land) within the mission area. The mission completion time and cost in terms of the distance traveled by the team was measured.

The CBR and runtime-CNP architecture was compared against a control. This baseline system used the same mission plan as generated by the CBR planner. It did not, however, use CNP-based task allocation. Instead, when a mine was discovered, a random robot was assigned the task of neutralizing it. Data pertaining to mission completion time and the cost in terms of the distance traveled by the team was collected over fifty runs for team sizes ranging from 2 to 6 robots.

In the second experiment, both systems were compared within a target interception scenario. In a target interception mission, a robot must overtake a mobile surface vessel launched from a near by pier (figure 2). The vessel interception experiments used a highly heterogeneous team of unmanned underwater vehicles (UUV), unmanned aerial vehicles (UAV), and unmanned surface vehicles (USV). Each vehicle type had differing velocity and fuel consumption characteristics.

Similarly to the NMC experiment, two sets of trials were conducted; the first with the system utilizing the CBR and runtime-CNP system, the second with the control. In the

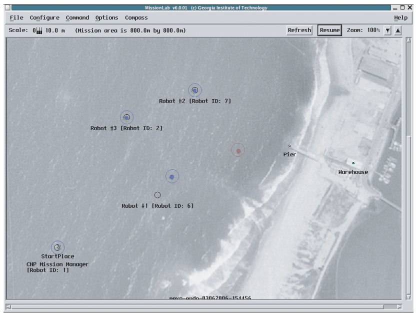


Fig. 2. Mission area used for interception scenario. Enemy vessel originates from the location marked pier.

TABLE I
TEAM MAKEUP FOR CBR AND RUNTIME-CNP TARGET INTERCEPTION EXPERIMENT.

Team Size	Team Composition
2 Robots	UUV, USV
3 Robots	UUV, USV, UAV
4 Robots	UUVx2, USV, UAV
5 Robots	UUVx2, USVx2, UAV
6 Robots	UUVx2, USVx2, UAVx2

TABLE II
MAXIMUM VELOCITY AND RANGE FOR ROBOT TYPES IN INTERCEPT SCENARIO

Robot Type	Maximum Velocity	Maximum Range
UUV	2.5 m/s	90km
USV	20 m/s	1152km
UAV	20 m/s	144km

TABLE III
FAULTS USED IN NMC SCENARIOS

Set	Faults
1	At $t = 30$, USV 1 velocity reduced by 50% At $t = 60$, USV 2 velocity reduced by 75%
2	At $t = 30$, USV 1 velocity reduced by 75% At $t = 70$, USV 1 velocity increase by 400% At $t = 90$, USV 3 velocity reduced by 25%
3	At $t = 30$, USV 1 fails At $t = 60$, USV 2 velocity reduced by 75%
4	At $t = 30$, USV 1 fails At $t = 70$, USV 2 fails

interception mission, the command and control vehicle, representing a distant surveillance aircraft, would generate a call for proposals detailing the target's last known position, heading, and velocity. When responding to this call for proposals, the team members first accept or reject it based on their ability to intercept the target. If the call for proposals is accepted, the bid submitted is calculated for robot n as $\frac{range_n - dist(robot_n, i)}{range_n}$, where i is the interception point. This allows the team to minimize the percentage of fuel used by the interceptor and thus maximize the ability of the team to respond to future goals. For the control, task allocation is decided by randomly selecting a robot from the set of robots that accept the call for proposals.

This experiment was repeated with teams ranging from 2 to 6 robots in random positions in the mission area for 50 trials in each configuration. The team composition for each of these experiments is shown in Table I, while the maximum velocity and maximum range for each of the robot types is shown in Table II. Data was collected pertaining to mission execution time and fuel consumption of the team.

2) Static Task Allocation Experiments: A second series of experiments was conducted in order to measure the ability of the CBR and runtime-CNP system to respond to changing conditions during mission execution. These experiments were conducted to examine the hypothesis that the CBR and runtime-CNP system will afford significant advantages over systems without task reassessment and reallocation, especially in dynamic and potentially hostile missions.

Two sets of experiments were conducted to test this hypothesis. The first set of experiments reexamines the NMC scenario. Similar in setup to the first NMC experiment, a team of four identical USVs are tasked with eliminating a number of mines within a litoral environment. During the execution of the mission, however, a series of faults were injected to abstractly simulate a variety of scenarios that may occur during mission execution. These faults, occurring at set times within the mission, fall into two categories: partial and full. Partial faults are meant to abstractly represent any situation that may affect the velocity of the robot. Examples of a partial faults may be rough weather or partial motor failure. Full faults, on the other hand, represent situations that prevent the robot from accomplishing the mission. These can include manipulator failure in the case of an NMC scenario or total motor failure.

Four trials of 10 runs were executed. Each trial introduces a set of two to three faults during each run. Two trials introduce partial faults while the other two trials introduce full faults. Table III shows the faults used in each of the four trials. Each set of the four trials were repeated twice, once with the CBR and runtime-CNP system and once with the baseline system. When a fault occurs to a robot within the CBR and runtime-CNP system, it will renege its contract if it is executing a task and request for the task to be reallocated (possibly to the same robot). The baseline system only supports static allocation. That is, once a robot has been awarded a contract for a task, it will execute the task until the task is complete. The heuristic used to calculate bids for the robots in both system was $b = \frac{v_n}{dist(robot_n, mine)}$, where v_n is the velocity of robot n. Performance was measured for both the CBR and runtime-CNP system and the static allocation system by measuring mission success rates and mission execution time.

In the final set of experiments, a target tracking scenario was examined. In this scenario, a target object is transported from a location on the mainland, across a channel, to an island where it is then taken by air out of the mission area (figure

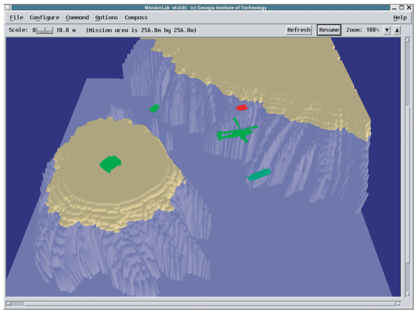


Fig. 3. Mission area used for target tracking scenario. Target begins on the mainland depicted on the top left of the mission area, travels to the island, and then flies from the mission area.

TABLE IV
ROBOT CHARACTERISTICS FOR TRACKING SCENARIO

Robot Type	Maximum Velocity	Mobility	Stealth
UUV	5 m/s	Sea	Yes
UGV	10 m/s	Land	No
USV	10 m/s	Sea	No
UAV	15 m/s	Air, Sea, Land	No

3). The team deployed in the mission area is responsible for tracking this object by following the transportation vehicle in a stealthy manner when possible. A robot that is stealthy in this scenario, means that the robot has low visibility (e.g. a UUV). The component tasks within this scenario are a series of tracking tasks based on the current mobility type (land, sea, or air) of the target, the target's current position, and stealth requirements. A team of four robots was used in this scenario: a UUV, UGV, UAV, and USV. Each vehicle has differing velocities, mobility, and stealth characteristics. These characteristics are summarized in table IV. The mobility characteristics of each vehicle determine what kind of targets can be tracked. (i.e. a UUV can track other underwater vessels or surface vessels). A robot's attempts to track a target outside of its mobility value results in failure and a potential reneging on its contract for that task.

During task allocation, each robot's bid for the task was calculated as $\frac{v_n}{dist(robot_n,target)}$, where v_n is the velocity of robot n. In addition, the bid is halved if the robot is not considered stealthy. Similar to the NMC scenario, four trials of 10 runs were executed for both the CBR and runtime-CNP system and the baseline. Two of the four contained only partial faults while the other two sets contained full faults. Table V shows the faults used in each of the four trials. In each trial, team members were placed in a random, valid location in the mission area. Mission performance was measured as mission success rates. In this case, a mission is considered successful if the target is actively tracked by a team member at least 50 percent of the mission duration.

TABLE V
FAULTS USED IN TARGET TRACKING SCENARIOS

Set	Faults
1	At $t = 30$, UAV velocity reduced by 90% At $t = 60$, USV velocity increased by 50%
2	At $t = 20$, UUV velocity reduced by 50% At $t = 40$, UAV velocity reduced by 40% At $t = 60$, USV velocity reduced by 25%
3	At $t = 30$, UUV fails At $t = 45$, UAV velocity reduced by 50%
4	At $t = 20$, UAV fails At $t = 40$, USV velocity reduced by 50%

B. CBR and Premission-CNP Design Evaluation

We recently reported [11] on the use of Goals, Operators, Methods, and Selection rules modeling (GOMS)[12][13] to evaluate the usability of both the underlying base system as well as to evaluate the impact of the CBR and premission-CNP design. In this earlier research a series of GOMS models was developed to characterize the base MissionLab system in addition to the system augmented with the premission CBR-CNP. GOMS models are a technique used in interface modeling to explicitly represent the knowledge users utilize to accomplish a particular goal with a user interface. To generate the GOMS model, a user interface analyst details the particular goals, operators, methods, and selection rules a user must follow to accomplish a task with the interface.

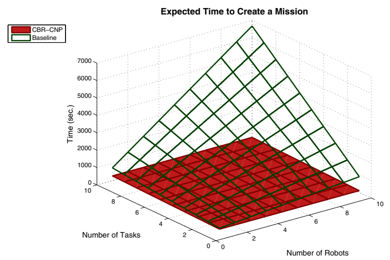
In a GOMS model, a goal represents a self-contained set of actions to accomplish a particular task. For mission specification as described here, a goal might be to add a new behavior to the robot's FSA. Operators represent primitive actions within the interface such as moving the mouse or clicking a button. Methods represent a sequence of operators that accomplish a goal.

Once models of both the base system and the CBR and premission-CNP system have been created, they allow for the verification, comparison, and quantization of a number of interface characteristics[1].

Through the analysis of these GOMS models, it was found that the mission creation time using the base system could be approximated by the function: $t(n, \tau) = A + Bn + 2Cn\tau$, where n represents the number of robots in the mission, τ represents the number of component tasks in the mission, and A, B, and C represent constant time factors within the model such as mouse clicks or cursor movement.

In contrast, the premission CBR/CNP-based system was found to have mission generation time approximated by the function: $t(\tau) = D + E\tau$, where τ is the number of component tasks and D and E are constant time factors in the model. From the analysis of the base system, it can be seen that for a given value of τ, the time required for mission creation increases linearly with the number of robots for the base system. When utilizing the CBR and premission-CNP design,

[1]The full models generated in the GOMS evaluation can be found at *http://www.cc.gatech.edu/ai/robot-lab/onraofnc/data/goms-2005/*.



cp

Fig. 4. Predicted cost for specifying a mission with a variable number of tasks and robots using $A = 78.3, B = 26.7, C = 34.8, D = 85.3, E = 24.3$ as calculated from the models.

however, mission creation time is constant with respect to the number of robots in the mission. Figure 4 depicts the predicted cost for generating missions using both systems for a variable number of tasks and team sizes. A more thorough discussion of these models, results, and additional verification of these models can be found in [11].

IV. Results

A. Random Allocation Experiments

Results of the naval mine countermeasures mission are depicted in Figures 5 and 6. The cost as measured via the distance traveled by team using the CBR and runtime-CNP architecture was found to be significantly less than that of the control for all team sizes. In most trials the CBR-CNP controlled team traveled approximately 50% less for all team sizes. In this experiment, the distance traveled by the control as well as the time spent executing the mission usually increases as team-size increases. In the case of the CBR/CNP-based team, however, an increase in team size leads to more efficient usage of the team as measured by the total distance traveled by the team. This result highlights how the importance of proper task allocation increases with the size of the team.

In the vessel interception experiments, a similar trend can be seen. Figure 7 depicts the time until mission completion. For all team sizes, the CBR/CNP-based team completes the mission in significantly less time than the control. The team's fuel consumption, as depicted in Figure 8, also shows that the CBR and runtime-CNP design performs significantly better than the control. Both interception results and the NMC results support our hypothesis, though the increase in performance with team size is only marginal.

B. Static Allocation Experiments

Figures 9 and 10 depict the results of the NMC scenario comparing the runtime system against a control only capable of static task allocation. The percentage of successful missions shows the importance of task reallocation when robot failure occurs. The baseline system, unable to handle such failures, completed only 30 percent of the trials. The CBR and runtime CNP architecture also performed the task significantly faster then the baseline system in these trials. In the trials in which only partial failure occurred, however, performance of both

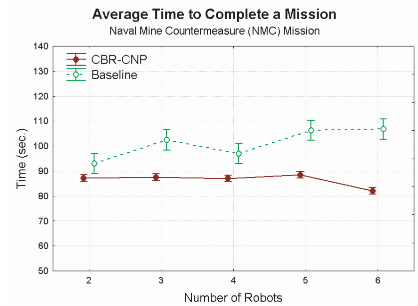


Fig. 5. a) Average time until removal of all mines for the CBR and runtime-CNP design vs. control. Error bars indicate 95% confidence intervals.

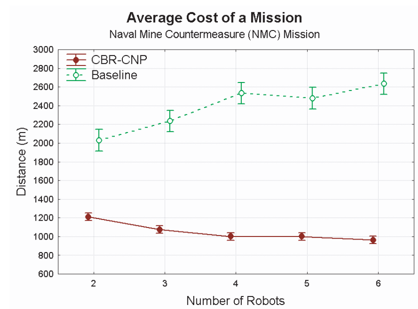


Fig. 6. Average cost as measured by distance traveled during the mission for the CBR and runtime-CNP design vs. control. Error bars indicate 95% confidence intervals.

systems was similar. Both the baseline and the CBR and runtime-CNP system were able to complete 100 percent of the these trials. In addition, both systems performed equally well in the trials with partial faults. This would seem to indicate that, at least in the scenarios tested, the time taken to initiate the auctions and reallocate the tasks (around 5 timesteps in these experiments) negates the performance gains afforded by reallocation when total robot failure does not occur.

The results of the final experiment, the target tracking scenario, are summarized in figure 11. The performance of the the CBR and runtime-CNP system, measured as the percentage of successful missions executed, exceeded that of the lesioned system for all trials. The CBR and runtime-CNP system provided the means in with which to adapt rapidly changing constraints of the tracking scenario mission, supporting our stated hypothesis. The baseline system with static task allocation, on the otherhand, more often resulted in the robots attempting to perform tasks no longer achievable (i.e. tracking a UGV with a UUV).

V. Conclusions

This work presented the evaluation of two approaches for combining case-based reasoning mission specification with a contract net protocol-based task allocation system. In the CBR and runtime-CNP architecture, a CBR planner generates mission plans for each robot that implements runtime task allocation as well as plans for accomplishing any awarded

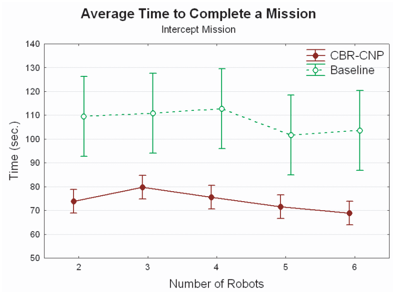


Fig. 7. Average time until interception of target vessel for the CBR and runtime-CNP design vs. control. Error bars indicate 95% confidence intervals.

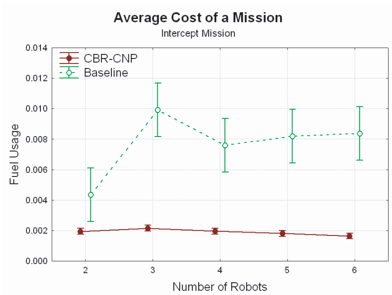


Fig. 8. Average cost as measured by percentage of fuel consumed by the team for CBR and runtime-CNP design vs. control. Error bars indicate 95% confidence intervals.

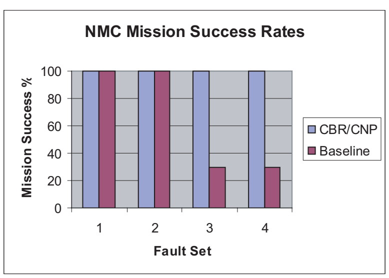


Fig. 9. Mission success rates for the NMC scenario for each of the four sets of faults.

tasks. This architecture was evaluated via simulation in three naval scenarios. In this evaluation the CBR and runtime-CNP evaluation has been shown to provide significant improvements over the static or random allocation strategies. In addition, the results of these experiments show that the runtime architecture is effective even in potentially adverserial conditions. Finally, a brief overview of earlier work [11] demonstrating the merits of the CBR and premission-CNP architecture via a GOMS usability analysis was provided.

ACKNOWLEDGMENT

This research was funded as part of NAVAIR's Intelligent Autonomy program.

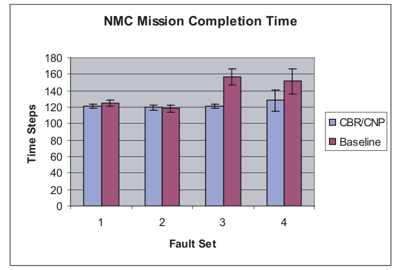


Fig. 10. Average amount of time required to complete the NMC scenario for each of the four sets of faults. Error bars indicate 95% confidence intervals.

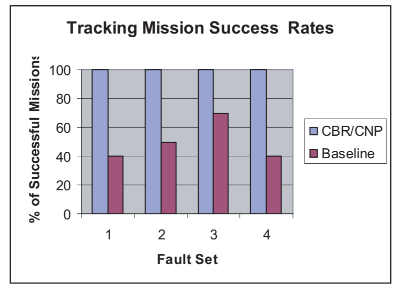


Fig. 11. Mission success rates for the target tracking experiments. A mission is considered successful if the target is actively tracked for at least 50 percent of the mission duration.

REFERENCES

[1] R. Smith, "The contract net protocol: High-level communication and control in a distributed problem solver," *IEEE Transactions on Computers*, vol. 29, no. 12, 1980.
[2] P. Ulam, Y. Endo, A. Wagner, and R. Arkin, "Integrated mission specification and task allocation for robot teams - Part 1: Design and implementation," under Review.
[3] D. MacKenzie, R. Arkin, and J. Cameron, "Multiagent mission specification and execution," *Autonomous Robots*, vol. 4, no. 1, pp. 29–52, 1997.
[4] Y. Endo, D. MacKenzie, and R. Arkin, "Usability evaluation of high-level user assistance for robot mission specification," *IEEE Transactions of Systems, Man, and Cybernetics*, vol. 34, pp. 168–180, 2004.
[5] B. Brumitt and A. Stentz, "GRAMMPS: A generalized mission planner for multiple mobile robots," in *Proceedings of the ICRA*, 1998.
[6] C. Le Pape, "A combination of centralized and distributed methods for multi-agent planning and scheduling," in *Proceedings of ICRA*, 1990, pp. 488–493.
[7] S. Botelho and R. Alami, "M+: A scheme for multi-robot cooperation through negotiated task allocation and achievement," in *Proceedings of ICRA*, 1999, pp. 1234–1239.
[8] M. Dias, R. Zlot, N. Kalra, and A. Stentz, "Market-based multirobot coordination: A survey and analysis," Carnegie Mellon University, Tech. Rep. CMU-RI-TR-05-13, 2005.
[9] B. Gerkey and M. Mararic, "A formal analysis and taxonomy of task allocation in multi-robot systems," *International Journal of Robotics Research*, vol. 23, no. 9, pp. 939–954, 2004.
[10] B. Gerkey and M. Mataric, "Sold!: Auction methods for multi-robot coordination," *IEEE Transactions of Robotics and Automation*, vol. 18, no. 5, pp. 758–768, 2002.
[11] A. Wagner, Y. Endo, P. Ulam, and R. Arkin, "Multi-robot user interface modeling," in *Proceedings of DARS 7*, M. Gini and R. Voyles, Eds. Springer, 2006, pp. 237–248.
[12] D. Kieras, "A guide to GOMS task analysis," University of Michigan, Tech. Rep., 1996.
[13] B. John and D. Kieras, "The GOMS family of user interface analysis techniques: Comparision and contrast," *ACM Transactions on Computer-Human Interaction*, vol. 3, no. 4, 1996.