Proceedings of the 2007 IEEE Symposium on Artificial Life (CI-ALife 2007)

Social, Physical, and Computational Tradeoffs in Collaborative Multi-agent Territory Exploration Tasks

Paul Schermerhorn and Matthias Scheutz
Artificial Intelligence and Robotics Laboratory
Department of Computer Science and Engineering
University of Notre Dame
Notre Dame, IN 46556
{pscherm1,mscheutz}@nd.edu

Abstract— **The performance of embodied multi-agent systems depends, in addition to the agent architectures of the employed agents, on their physical characteristics (e.g., sensory range, speed, etc.) and group properties (e.g., number of agents, types of agents, etc.). Consequently, it is difficult to evaluate the performance of a multi-agent system based on the performance of an agent architecture alone, even in homogeneous teams.**

In this paper, we propose a method for analyzing the performance of multi-agent systems based on the notion of "performance-cost-tradeoff," which attempts to determine the relations among different cost-dimensions by performing a performance sampling of these dimensions and comparing them relative to their associated costs. Specifically, we investigate the performance-cost tradeoffs of four candidate architectures for a *multi-agent territory exploration* task in which a group of agents is required to visit a set of checkpoints randomly placed in an environment in the shortest time possible. Performance tradeoffs between three dimensions (sensory range, group size, and prediction) are then used to illustrate the cost-benefit analyses performed to determine the best agent configurations for different practical settings.

I. Introduction

The performance of an agent architecture critically depends on the *computational resources* that an agent has available for implementing an *agent function* [11], which specifies the mapping from sequences of percepts to actions of the agents. A solution that is optimal for a Turing machine might not be *implementable* in a given architecture due to memory constraints; only "sub-optimal" solutions can be feasibly achieved. In embodied agents, additional limiting factors come into play that could restrict the performance of the agent such as the cost of obtaining a particular percept (e.g., the cost of a having a particular sensor that can provide the information necessary to produce a percept or the cost of having a sufficiently long sensory range to be able to have the percept at a particular time). Orthogonally, in multi-agent systems the performance of the whole system might depend, in addition to the computational resources of each individual agent, on the number of participating agents. Most web-based search agents, for example, can retrieve information more quickly given that they can search multiple sites in parallel. Most importantly, in embodied multi-agent systems (i.e., multi-robot systems) the system performance will be a function of the computational resources and physical characteristics of each agent, as well as the properties pertaining to the team as a whole. A team of robots for planetary exploration, for example, will, among other things, be subject to weight constraints (i.e., the sum of the weight of all robots), given that it needs to be transported to the remote planet via a rocket. Consequently, the question arises of how to evaluate the performance of a multi-agent system in terms of individual agent architectures when other factors might crucially affect its performance?

In this paper, we present a method of analyzing the performance of multi-agent systems based on the notion of "performance-cost tradeoff," which attempts to determine the relations among different cost dimensions by conducting a performance sampling of these dimensions and comparing them relative to their associated costs. Specifically, we investigate the performance-cost tradeoffs of four candidate architectures for a *multi-agent territory exploration* task in which a group of agents is required to visit a set of checkpoints randomly placed in an environment in the shortest time possible. *Reactive* and *deliberative* architectures are introduced and serve as bases for exploring performance tradeoffs along three dimensions. The first dimension is *physical*: extension of sensory range. Increasing the agent's ability to detect remote checkpoints is an effective way of improving performance without incurring costly computation and communication costs. However, as the sensory range increases, the cost also increases, hence at some point increasing the range will not be worth the additional cost. The second dimension is *social*: group size. In many cases, increasing the number of agents working on a problem will increase performance (although there will be a point at which increasing group size fails to increase, and may even decrease, performance). Once again, however, increasing the number of agents increases cost, and eventually will not be beneficial. The third dimension is *architectural*: modifications to the architecture to facilitate agent coordination. The specific modification examined here employs a simple predictive mechanism to allow agents to select actions that are less likely to be in conflict with other agents' actions. This reduces the amount of redundant work performed by individual agents, making their overall efficiency on the group task better (i.e., reducing the time it takes the group to

1-4244-0701-X/07/$20.00 ©2007 IEEE

complete the task). We will show that the relationship among these three dimensions in terms of the performance of the agent system is not *apriori* obvious and that samplings of the performance space given by the three dimensions via extensive simulation studies is a useful method to obtain criteria for deciding which configuration to use in a given practical setting.

The remainder of the paper proceeds as follows: we first define the task that forms the context in which the architectures will be evaluated. We then describe the base architectures as well as the architectural enhancements tested in our experiments. The experimental design and setup are given, and results are presented. This is followed by an example of the evaluation of the architectures in light of the experimental results using hypothetical values for component costs. Finally, we conclude with some discussion of the implications of this research.

II. Methods

The *multi-agent territory exploration* (MATE) task requires a group of A agents to visit C checkpoints in an environment while avoiding B obstacles. Agents and checkpoints are randomly placed, and the goal is to minimize the total time required to visit all C checkpoints, rather than to maximize the number of checkpoints visited by any single agent. Many variants of MATE tasks have been investigated in the literature [8], [5], [6], [15], [2]. E.g., the planetary scenario mentioned above is an instance where a number of robots is dropped onto another planet for the purpose of visiting a set of geologically interesting rock samples. The robots would need to visit their targets as quickly as possible in order to complete their mission before running out of fuel. Furthermore, landing robots on distant planets is an inexact science, with locations specifiable only in the most general terms, and catastrophic failures a distinct possibility. It is, therefore, impossible to create a static plan ahead of time, ready to execute upon landing. Requiring the rovers to compute the optimal solution upon landing is impractical (or impossible), and the conditions may change during the course of the task (e.g., a rover may fail, leaving the others to complete its mission).

While the optimal solution in a MATE task can be determined by exhaustively searching all assignments of checkpoints to agents and all visitation orders within each assignment, it is typically computationally intractable (given that the problem of finding a solution in MATE tasks is NP-complete).[1] Furthermore, such an approach is a static, offline solution; if the environment is subject to change during the course of the task, the expense of recomputing the optimal solution must be incurred again.

[1]Note that MATE is similar in some ways to the multi-Traveling Salesman Problems (MTSP), in which some member of a sales team is required to visit each city. While good approximations exist for a particular instance of MTSP, the Delivery Scheduling Problem [7], [4], [17], [1], [9], [16], MTSP variants require agents to return to their initial positions, which is not a requirement of MATE. Hence, none of the MTSP approximations is an exact fit for the MATE task.

What is needed is a flexible, online solution that inexpensively produces good (even if not optimal) behavior at a low cost. Simple reactive agents, for example, can be quite successful in MATE tasks at a very low cost by employing a straightforward greedy strategy: always visit the closest perceivable checkpoint, otherwise perform a random walk until a checkpoint is perceived. This strategy can be improved upon by adding deliberative capacities that allow the agent to plan efficient routes around obstacles and to remember the locations of previously sensed checkpoints. The added performance for deliberative agents may be worth the additional cost in some environments.

A. Relative Performance

We begin the analysis of agent performance with a comparison of *absolute performance* in Section II-C. We start with a base agent model, which is modified to include some new feature F. The performance of agents with and without F are then compared. However, absolute performance provides only the starting point for any full performance analysis. The costs of the compared systems are factored in to acquire *relative performance*. When comparing results (agents with F (A_{F^+}) vs. agents without F (A_{F^-})), there are three possible outcomes:

1) $Perf_{abs}(A_{F^+}) \leq Perf_{abs}(A_{F^-})$**.** In this case, we know that agents with F have worse relative performance, because there will always be some cost associated with any feature. If there is no absolute performance benefit to offset the cost, there can be no relative performance benefit.
2) $Perf_{abs}(A_{F^+}) > Perf_{abs}(A_{F^-})$**.** In this case, there is the possibility of a relative performance benefit to F. If such a result were found, it would be necessary to undertake a systematic analysis of the cost of the new feature in order to determine whether there is a relative performance advantage to F.
3) $Perf_{abs}(A_{F^+}) \gg Perf_{abs}(A_{F^-})$**.** In this case, it is very likely that F *does* provide a relative performance benefit, because it is very likely that the large benefits will outweigh the costs.

Of these three cases, the first and the third allow one to make a judgement about relative performance based only on absolute performance. The second case is the interesting one, because it is here that cost must be examined. At one extreme of this interval, there is only a small performance advantage, making a relative performance advantage very unlikely. At the other extreme, it is becoming more likely that a relative performance advantage exists, as the benefits begin to dominate the costs. In the middle, there are likely to be some cases in which there is no clear answer (e.g., it may be beneficial for one set of inputs but not for another).

In multi-agent exploration tasks, costs can vary along multiple dimensions. The physical cost of the multi-agent system as a whole is one. This cost will be affected largely by the number of agents in the system, and may manifest itself in many ways. If, for example, agents are expected to return to a "home base" for recharging, the capacity of

the charging station component of the system will need to increase as group size increases. In the case of extraplanetary landers, a prominent concern will be the weight of the multi-agent system as a whole, because the transport vehicle will need to scale its capacity with any increase in group size. We examine the impact of group size in the experiments below.

The physical costs of individual agent systems will also be a factor. Individual weight is a factor, with heavier agents requiring greater energy for maneuvering. Similarly for agent speed: faster agents require not only more energy to move, but also potentially more costly components to allow the higher speed, all else being equal. Long sensory range can help to achieve high absolute performance but can be very costly, as signals tend to drop in intensity at a quadratic rate, requiring quadratic cost to maintain a constant signal level. We examine the effect of changing sensory range below.

Finally, there is the cost associated with the architecture (i.e., control costs). Complexity in the control system can be traded for better absolute performance. In some situations, simple reactive architectures will perform well, while in others the ability to remember and plan are a virtual necessity. Increases in control system complexity are accompanied by increases in energy use, heat production, and in many cases weight, as more processing capacity may be needed. We examine two architectural enhancements of different complexity below.

The technique of comparing relative performance (i.e., performance with cost factored in) applies not only to MATE and MATE-like tasks, but to virtually any analysis of agent performance where a decision must be made among several parameters or components that will affect performance. For example, the designer of a software agent may need to decide between multiple search methods. An exhaustive search will provide the optimal solution, but at a high cost in terms of complexity. There are some cases in which some heuristic search method can provide sufficiently good results at significantly reduced cost.

B. Reactive and Deliberative Agent Architectures

We employ two basic architectures in this study. The first is a reactive architecture that maps perceptions onto actions directly. All reactive agents process sensory information and produce behavioral responses using a motor schema-based approach [3]. Let $Ent = \{c, b, a\}$ be an index set of the three types of objects: *checkpoints*, *obstacles* and *agents*. For each object type Ent, a force vector F_t is computed, which is the sum, scaled by $1/|v|^2$, of all vectors v from the agent to the objects of type t within the respective sensory range, where $|v|$ is the length of vector v. These *perceptual schemas* are mapped into motor space by the transformation function

$$T(x) = \sum_{t \in Ent} g_t \cdot F_t(x) \tag{1}$$

where the g_t are the respective gain values of the perceptual schemes. The gain values simply scale the effect of sensory input, providing a means by which to prioritize certain inputs (e.g., if visiting checkpoints is especially important, the checkpoint gain value could be higher than the agent gain value, so that sensing a checkpoint has a greater impact on the direction chosen than sensing other agents). These gains are initialized to values determined to be reasonable via a series of experiments, and are kept constant throughout the life of a reactive agent.

A collision detection mechanism invokes an agent's retreat reflex whenever it detects an impending collision with an obstacle or another agent (all collisions are fatal). The reflex works by inserting a very strong vector leading away from the site of the near collision. This vector is included for a random number of cycles between 5 and 15, and has the effect of moving the agent directly away from the object or agent. The reflex works well in most cases, although it is possible to fail in some situations (e.g., it may be possible to retreat into another obstacle in some circumstances).

Finally, to prevent agents from missing the checkpoints altogether and diverging indefinitely, two additional behaviors are included. The first is a *random walk* mechanism that changes an agent's heading when it has not recently sensed any checkpoints. The second has agents "bounce off" the edge of the environment as if it were a wall. These simple mechanisms greatly reduce the number of task failures.

Reactive agents always behave in the same way, given that their gain values are *constants*: their positive g_c makes them employ a greedy visitation strategy (most of the time a "visit nearest" strategy [14]), whereas their negative g_b and g_a values make them avoid obstacles and other agents. The effect of g_a on the reactive agents' behavior is to establish implicitly a "ranking" of who gets to visit a checkpoint first if multiple agents attempt to visit the same checkpoint: whoever is closest will be more strongly attracted to the checkpoint than repelled by the other agents, and hence be able to get to visit the checkpoint, whereas the other agents will be repelled more by the presence of agents than they are attracted to the checkpoint, and hence will move away. In a sense, g_a implements a simple "coordination" strategy, if only one that is "negatively" determined.

The second basic architecture employed in this study is deliberative. Deliberative agents have several components that allow them to manipulate representations of checkpoints in the environment. Most importantly they have a route planner that can determine which checkpoint is closest to them and how they can best get to it. It is first and foremost this ability of being able to represent entities in the environment that opens up further possibilities such as storing and retrieving representations, using them in planning and plan execution, etc. None of these possibilities is available to reactive agents, which have access to sensed objects only in a holistic manner (via agglomerated force vectors).

The planner of the deliberative agents (based on a simplified version of the A^*_ϵ algorithm [10]) is given a list of checkpoints known to the agent (i.e., stored in the agent's memory), and returns a *plan*—a sequence of headings and distances representing a clear path to the nearest reachable checkpoint. The plan is then passed to a *plan execution*

mechanism, which ensures that plan steps are executed. When other agents cross a deliberative agent's route and the reflex is triggered, "re-planning" is initiated, and the agent will continue by executing the new plan. Re-planning is also performed if the checkpoint chosen by the agent has been visited by another agent in the meantime. A further difference between deliberative and reactive agents is that, while the schema-based mechanism of the reactive agents will not pick out the most direct route to a checkpoint (because of the influence of other checkpoints and agents), and may even move *away* from the nearest goal checkpoint (because of a cluster of objects further away in the opposite direction, or a cluster of agents in the direction of the nearest checkpoint), deliberative agents will find the nearest checkpoint and plan a route directly to it (while avoiding other agents and obstacles), thus saving time and energy. In the event that there are no checkpoints to be visited in a deliberative agent's sensory range or in its memory, it reverts to reactive behavior, foraging for checkpoints using the random walk mechanism described above.

C. Architectural Enhancements

We enhanced the basic architectures in three ways, one a physical modification, one a social modification, and the other a control-system modification. The first enhancement is simply to extend the sensory range R. All agents share the same sensor configuration, including smell, vision, and sonar. Reactive agents employ smell to create the vector for checkpoints to be visited and sonar to create the agent and obstacle vectors. Note that smell and sonar do not provide the location of any individual agent, but rather contribute to the summed influence of all checkpoints (in the case of smell). Normal reactive agents do not use vision, but the prediction extension described below uses it to locate individual agents and checkpoints. Deliberative agents use smell and sonar in the same way reactive agents do, although in most cases the deliberative layer suppresses their effects in order to execute a plan; only when there is no checkpoint in sensory range do the reactive mechanisms take effect. Vision is used by deliberative agents to locate individual checkpoints, agents, and obstacles in order to store them in memory and generate plans for visiting checkpoints.

Extending the sensory range of the agents is an intuitive way to improve their performance. For one thing, the more information an agent has about its surroundings, the better its decisions can be. Furthermore, with very small sensory range, it will often be the case that agents simply do not perceive *any* checkpoints, making it impossible for them to target any checkpoint for visitation and requiring them to revert to the random walk behavior described above. However, sensory range increases with diminishing returns, especially as the range approaches the size of the environment, as later discussion will demonstrate.

The second enhancement is an increase in group size. This is another intuitive way to increase performance. Additional agents will increase the likelihood that some agent is near any individual checkpoint, reducing on average the time it takes for the checkpoint to be located. This increase in parallelism comes at a cost, however, and also will provide diminishing returns when group size is high.

The third enhancement we explore here is a primitive prediction mechanism. The goal of the mechanism is to prevent duplication of effort by trying to avoid more than one agent attempting to visit a single checkpoint. The prediction mechanism functions as a *perceptual filter* that makes "educated guesses" as to which checkpoints it would be fruitless to pursue and excluding those from influencing the agent's behavior. For each agent in the current agent's visual range, the closest checkpoint to that agent is located. If that checkpoint is closer to the other agent than to the current agent, it is excluded on the assumption that the other agent will attempt to visit it.

What is interesting about this prediction mechanism is that it directly benefits the individual agent to use it, however, to the extent that it works, it also contributes to the group's goal of visiting all checkpoints in the shortest possible time. The benefit to the individual is clear: if another agent is guaranteed to visit a checkpoint before current agent arrives, selecting that checkpoint as a target will lead to expense without the possibility of reward. Thus, it is in the best interest of the agent to ignore that checkpoint anyway. The emergent behavior produced is an implicit form of cooperation (i.e., the effect of the prediction mechanism is as though the agents are cooperating, even though from their perspective, they are operating in their own self-interest).

Adding the prediction mechanism to the base agent types results in four agent types: normal reactive, reactive-predictive, normal deliberative, and deliberative-predictive. Reactive-predictive agents prevent checkpoints identified as potentially targeted by other agents from being added to their perceptual schemas. Deliberative-predictive agents do not add those checkpoints to memory, preventing them from being identified as goals for the planner.

Note that this simple prediction is far from perfect. It will not always filter the checkpoints targeted by other agents. For example, if two other agents share the same closest checkpoint, one of them will (using the same prediction mechanism) filter out that checkpoint and pursue another checkpoint. Also, the mechanism may correctly filter out the closest checkpoint to another agent, but will not notice that the checkpoint the other agent will pursue after that should also be filtered because it, too, is closer to the other agent than to the current agent. Situations such as these will lead to duplication of effort even in predictive agents, but as we show below, even this imperfect prediction can improve performance substantially. More sophisticated prediction is, of course, possible, but would be significantly more costly in terms of computational resources.

III. Experiments and Results

In order to gauge whether increased sensory range or the prediction mechanism would improve performance (i.e., decrease the time to completion) for the MATE task, we conducted a series of experiments in simulation. The artificial

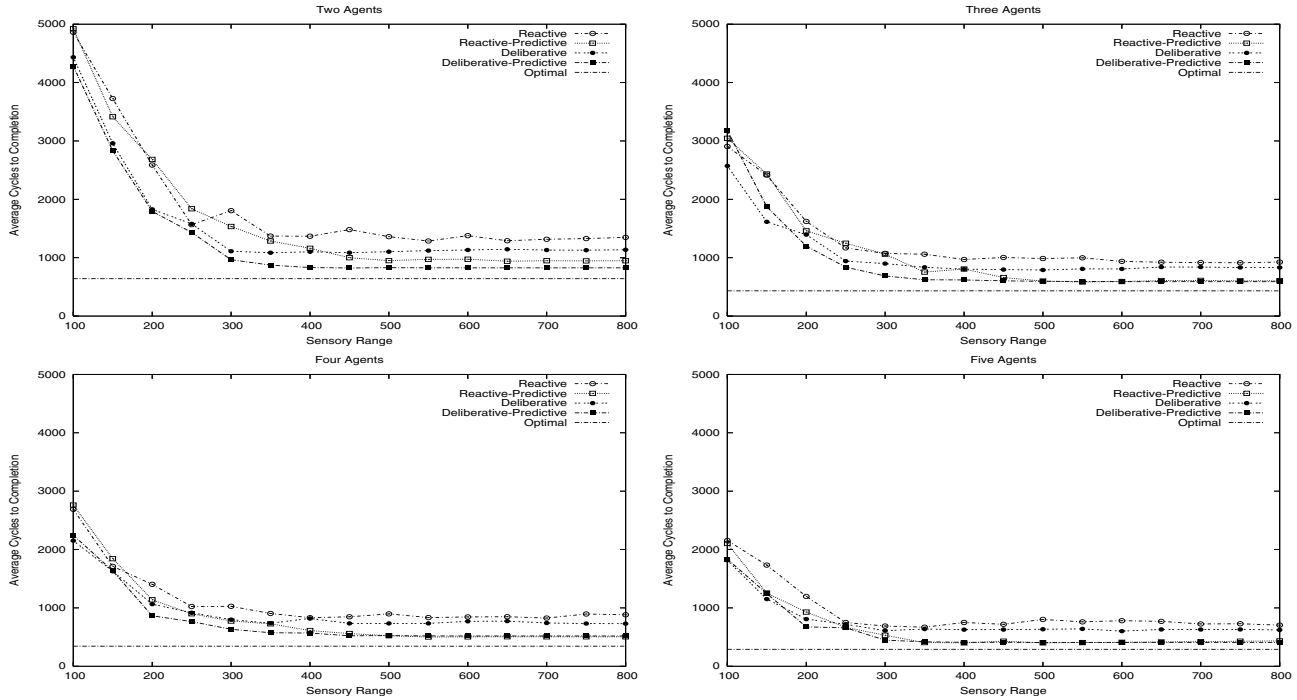

Fig. 1. Average performance of two- to five-agent experiments in five-obstacle environments for sensory ranges from 100 to 800.

life simulator SWAGES under development in our lab[2] was used as a platform in which implementations of the four architecture types (Reactive, Reactive-Predictive, Deliberative, and Deliberative-Predictive) were tested. The world in which the agents operate is 800 by 800, and is bounded. The number of agents (A) was varied from 2 to 5[3]. The MATE task was defined for ten checkpoints ($C = 10$), and was conducted in environments containing five obstacles ($B = 5$). Finally, the agents' sensory range (R) was varied from 100 to 800, with a step of 50.

Each experiment consists of 38 experimental runs using different randomly generated initial conditions. Each set of 38 initial placements of agents and checkpoints is used for all experiments with identical A, and all obstacle environments share the same positions for obstacles. This allows us to compare directly between agent types and sensory ranges. The results reported here are the average cycles to completion of each experiment set for each architectural configuration. In addition to these results, the graphs in Figure 1 depict the performance of the optimal solution for that group size to serve as an upper bound on performance. The performance of the optimal solution was calculated separately for each set of initial conditions via an exhaustive search of all possible assignments of checkpoints to agents and all permutations of order for each assignment. Unlimited sensory range is assumed in the optimal solution, making it the best possible performance on the task [13].

Figure 1 represents performance for 2 to 5 agents. Increasing sensory range increases performance quickly at first, then tapers off between 300 and 450, at which point increased sensory ranges do not contribute to better performance. All agent types see their performance rise and then level off, but at different sensory ranges. Performance of non-predictive types tends to level off at lower sensory range than performance of predictive types. This is an indication that predictive agents are better able to take advantage of extended sensory ranges. Alternatively, it could be argued that the other agent types utilize additional sensory range more efficiently, reaching their leveling-off point earlier than predictive agents. For non-predictive agent types, deliberative agents perform consistently better than reactive agents. This is a result of deliberative agents having the ability to plan efficient routes around obstacles, rather than having to take indirect routes around them as the reactive agents are forced to do by their schema-based approach. However, with the addition of the prediction mechanism, this difference is dependent on other factors. In two-agent groups (Figure 1, upper left), reactive-predictive agents approach, but never achieve, deliberative-predictive agent performance. For group sizes of 3, 4, and 5, however (remainder of Figure 1), reactive-predictive agents *do* achieve performance similar to deliberative-predictive agents, at higher sensory ranges. Moreover, as group size increases, the sensory range at which

[2]SWAGES is an agent-based simulation environment built with the SIMAGENT toolkit, freely available at http://www.nd.edu/~airolab/swages and http://www.cs.bham.ac.uk/research/poplog/newkit.tar.gz, respectively. It consists of a continuous, potentially unlimited two-dimensional surface populated with various kinds of spatially extended objects, in particular, different kinds of agents and resources they need as well as various kinds of static and moving obstacles.

[3]While we could have examined larger groups, group performance already begins to converge at $A = 5$ for MATE in these environments, making larger groups uninformative. For other tasks or environmental conditions, larger groups may be more appropriate.

the two kinds converge decreases.

To confirm the results depicted in Figure 1, a four-way 2x2x15x4 ANOVA was conducted on the dataset (not the averages), with *agent kind* (reactive and deliberative), *prediction* (without and with), *sensory range* (100 to 800), and *group size* (2 to 5) as dependent variables, and *cycles to completion* as the independent variable. Agent kind ($F(1, 8880) = 132.5605, p < 0.001$), prediction ($F(1, 8880) = 149.0721, p < 0.001$), sensory range ($F(14, 8880) = 69.0042, p < 0.001$), and group size ($F(3, 8880) = 532.6412, p < 0.001$) were all found to be highly significant main effects for cycles to completion. This indicates that each of the four independent variables has a significant effect on performance; each of these is to be expected. There were also highly significant two-way interactions between agent kind and sensory range ($F(14, 8880) = 4.5786, p < 0.001$), confirming that the significant performance differences between agent kinds are found at higher sensory ranges, between prediction and sensory range ($F(14, 8880) = 4.0314, p < 0.001$), confirming that performance differences between non-predictive and predictive agents are not significant at lower sensory ranges (i.e., when agents are less able to perceive other agents), between agent kind and group size ($F(3, 8880) = 9.9915, p < 0.001$), confirming that performance differences between reactive and deliberative agents are most pronounced in smaller groups, and between sensory range and group size ($F(42, 8880) = 20.0382, p < 0.001$), confirming that smaller groups benefit more from increased sensory range, at least when sensory range is low. There were no significant three-way or four-way interactions.

It is interesting that reactive-predictive agents outperform non-predictive deliberative agents, performing virtually identically to deliberative-predictive agents with sufficiently high sensory ranges. Also of note is the fact that performance of normal reactive agents can actually *decrease* at higher sensory ranges. This is attributable to the reactive agents' schema-based system: when they detect a remote checkpoint, they are attracted to it even if the vector is dominated by a closer checkpoint. This leads to an indirect trajectory to the near checkpoint. Reactive-predictive agents are less (but still somewhat) susceptible to this effect, as they will often ignore remote checkpoints due their proximity to other agents.

IV. Determining the Optimal Configuration

Finding the best configuration for the multi-agent system is essentially a multi-dimensional optimization problem, possibly with constraints (e.g., minimum absolute performance). We use the absolute performance results and cost assessments of each dimension to demonstrate the method of determining the best performance tradeoff between the control (architecture), physical (sensory range), and social (group size) dimensions. Note that the cost assessments used here are for expository purposes only; they are not intended to be accurate reflections of cost in any real system. Furthermore, the results of the following analysis are not intended to be taken as showing that one configuration is conclusively superior to all others; accurate cost assessments *would* be necessary to support such an empirical claim, unless the cost or performance differences were so large that the tradeoffs were obvious (e.g., when one agent configuration fails to achieve a minimum performance requirement). Even with accurate cost estimates, these analyses may be applicable only to the specific implementations under investigation (as such accurate cost figures are not possible for generic implementations of an architecture). The analysis here serves two purposes. First, it demonstrates the importance of including cost in performance analysis. When competing agents are evaluated based only on their absolute performance, important considerations are ignored. Second, it demonstrates a method of taking cost into account given (previously determined) cost parameters.

Before beginning the performance-cost analysis, it is sometimes useful to determine whether a high-level view of the cost and performance values for the candidate architectures can eliminate one or more options. In terms of computational (control) complexity, an ordering between the four agent types examined here looks like this: normal reactive agents have the lowest complexity ($O(|A| + |B| + |C|)$, the complexity of summing the vectors of all entities), followed by reactive-predictive agents ($O(|A| \cdot |C|)$, the complexity of the predictive mechanism, which must examine each checkpoint for each agent in the worst case). Deliberative agents have by far the highest complexity as a result of the exponential cost of the A^*_ϵ planner.

More complex computations will require more complex computational architectures, in turn leading to higher structural costs (i.e., costs associated with possessing and maintaining an architectural component). Given the significant difference in structural cost between deliberative and reactive architectures and the relatively similar performance on the MATE task (i.e., there is no significant difference in performance between predictive deliberative and reactive agent kinds for this task, and only a small, albeit significant, performance advantage for non-predictive deliberative vs. non-predictive reactive relative to the exponential cost of the deliberative planner), it will be reasonable in most cases to select a reactive agent architecture for the MATE task instead of a deliberative architecture.[4] Furthermore, reactive and reactive-predictive architectures are easy to implement on robots in the laboratory. The following discussion will, therefore, focus on the tradeoffs between reactive agents of both kinds with varying sensory ranges.

Determining the best architecture for a task will involve multiple considerations, including examining the performance to cost ratio for each architecture under consideration. For reactive and reactive-predictive architectures, given that they share the same base architecture, the structural cost will include some base cost b that is common to all configurations and includes the cost of some minimum sensory range. The

[4]However, the MATE task as defined here is very simple; there could be ecological niches in which more complex (e.g., deliberative) architectures will outperform their simpler opponents sufficiently to warrant their increased cost.

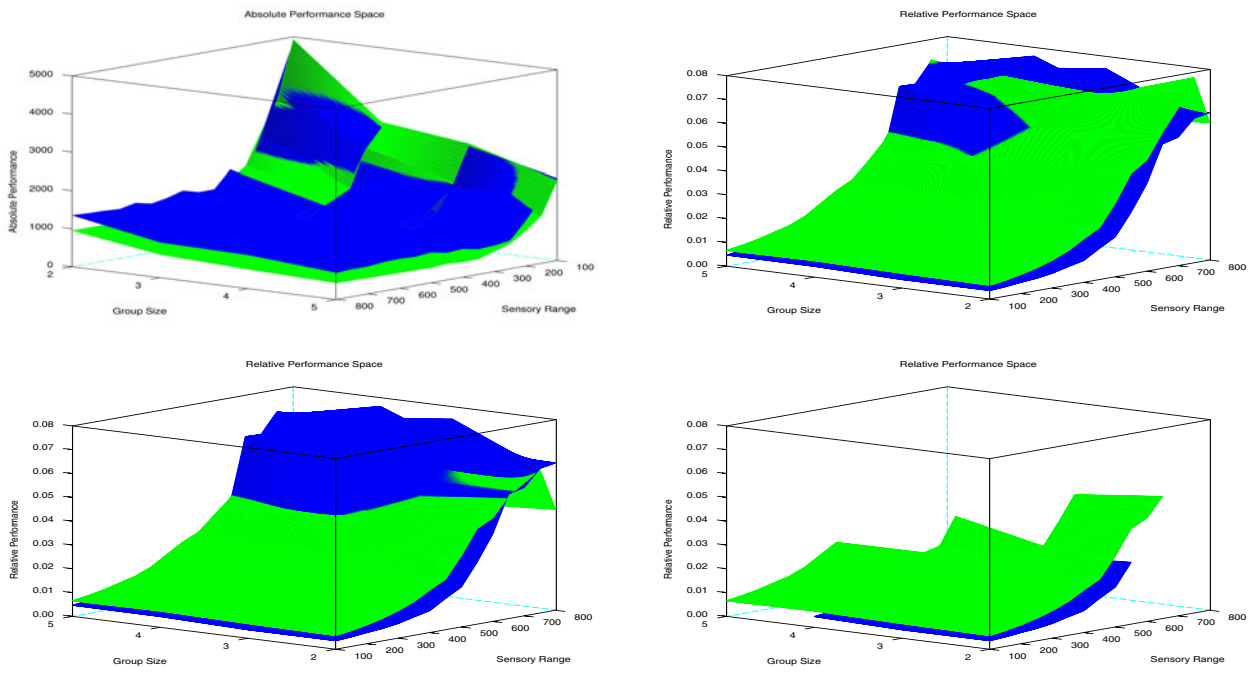

Fig. 2. **Upper left:** Absolute performance space (no cost). **Upper right:** Relative performance space when prediction cost is 1.5. **Lower left:** Relative performance space when prediction cost is 7.5. **Lower right:** Relative performance space when prediction cost is 7.5, constrained to 1000 cycles or less. Blue (darker) are normal reactive, green (lighter) are reactive-predictive.

cost of additional architectural components or enhancements can be expressed in terms of the base cost.

For example, the cost of increasing sensory range by some distance $range_{inc}$ could be expressed as some fraction of the base cost (e.g., $\frac{b}{r}$, where r is some constant representing the relationship between the base cost and sensor cost). Signal intensity drops quadratically with distance ($\frac{1}{dist^2}$), requiring a quadratic increase in cost to maintain a constant signal level, so the overall cost of increasing sensory range could be expressed as

$$C_{sensors}(range_{inc}) = \left(\frac{b}{r} \cdot range_{inc}\right)^2,$$

where $range_{inc} = range_{total} - range_{base}$ (i.e., the increase over the base sensory range). Thus, increasing the sensory range of an agent beyond its base range will increase its structural cost as a function of the amount of increase. The cost of the class of reactive architectures with various sensory ranges is then

$$C_{reactive}(range_{inc}) = b + \left(\frac{b}{r} \cdot range_{inc}\right)^2,$$

Adding the predictive mechanism to an architecture will also increase the structural cost. However, in this case, the additional cost is a constant amount, say, $b \cdot p$ (where p is some constant). That makes the cost of reactive-predictive agents $b + b \cdot p$. Coupling that with the potential for extended sensory ranges yields a structural cost function of

$$C_{reactive-predictive}(range_{inc}) = b + b \cdot p + \left(\frac{b}{r} \cdot range_{inc}\right)^2,$$

Consider, for example, the case in which $b = 15$, $r = 7.5$, and $p = \frac{1}{10}$. Furthermore, assume that sensory range increases by increments of 50, as in the experiments described above, and that the cost of the initial range of 100 is included in b. This makes the cost function for normal reactive agents $C_{reactive}(range_{total}) = 15 + \left(\frac{15}{7.5} \cdot (range_{total} - 100)\right)^2$. Similarly, reactive-predictive agents will incur structural costs of $C_{reactive-predictive}(range_{total}) = 15 + \frac{15}{10} + \left(\frac{15}{7.5} \cdot (range_{total} - 100)\right)^2$.

Figure 2 (upper right) plots the performance-cost ratios for reactive and reactive-predictive agents in the experiments described above using these assumed costs. In each case, the cost is multiplied by the number of agents working on the task. The scaled performance space can be used to determine the best sensory range for a given group size of a particular agent type (e.g., for normal reactive agents, the peak is at sensory range 150 for group size 2), or the best combination of group size and sensory range for that kind (4 and 150 for normal reactive agents). Determining the best overall configuration, including sensory range, group size, and agent type (reactive or reactive-predictive) is often a matter simply of finding the overall peak of the performance space. For these cost values, the best performance-cost ratio is found in reactive-predictive agents with groups of size 5 and sensory range 150. Compare this with the minimum cycles to completion in the unscaled performance space, which is found in reactive-predictive agents with group size of 5 and sensory range of 500. Absolute performance at $A = 5$, $R = 500$ is 401.026, while at $A = 5$, $R = 150$ it is 1254.76. This is a substantial performance difference, but the added cost of increasing the sensory range by 350 is sufficient to give shorter range a relative performance advantage. Normal reactive agents have an absolute performance of 1733.71 at $A = 5$, $R = 150$; the absolute performance advantage of

reactive-predictive agents is sufficient to overcome the added cost of the prediction mechanism.

As noted, the cost values used in the above example are fairly arbitrary, and not meant as an accurate indication of the relative costs of the respective components, but rather to illustrate the importance of including cost in performance analysis. Different cost assumptions will change the subsequent analysis. If, for example, the cost of the prediction mechanism is changed to $p = \frac{1}{2}$ (while the rest of the constants remain the same), the picture changes substantially (Figure 2, lower left). The additional cost of prediction is now sufficient to decrease the best predictive configuration's relative performance, which is still found at $A = 5$, $R = 150$, such that there are eleven non-predictive configurations with higher relative performance, the best of which is found at $A = 4$, $R = 150$.

In some cases, selecting the best performance-cost ratio will be too simplistic; there may be considerations other than cost that must be taken into account. So, while in the previous example non-predictive agents in groups of size 4 with sensory range of 150 had the best overall ratio, if we add the constraint that the task must complete in 1000 cycles or fewer, the outcome changes again. Figure 2 (lower right) depicts the remaining performance space, with non-qualifying points removed.

This type of analysis allows designers to choose the best combination of architectural features for a given task, making explicit the fact that raw performance does not tell the whole story, but cost must also be a consideration when selecting architectures and architectural components. Removing the prediction component, to use the example explored here, will lead to reduced cost, but the accompanying performance decrease may reduce the performance-cost ratio too much, making non-predictive agents worse than agents maintaining the component. Similarly, increasing or decreasing sensory range or group size will lead to changed performance-cost ratios, again potentially shifting the peak in the scaled performance space. Constraints (such as a minimum performance requirement) can be applied first, and then a performance space can be computed and the best configuration chosen.

V. Conclusion

In this paper we demonstrate the importance of including cost in agent evaluations and describe a method for comparing architectures using performance-cost analysis. To begin, we introduce two agent architectures for MATE tasks, reactive and deliberative, and two simple extensions to them. Predictably, increasing the sensory range yields increased performance in MATE, up to a point. Implicit cooperation emerges via the use of the primitive prediction mechanism tested here, and while its performance is not optimal, it increases performance significantly, allowing reactive-predictive agents to outperform nonpredictive deliberative agents and to perform on a par with deliberative-predictive agents. This is true even in environments containing obstacles, which previous research indicated favored deliberative agents in terms of raw performance [12].

To account for architectural costs, we also conducted multiple performance-cost analyses of reactive and reactive-predictive agents using hypothetical values for architectural parameter costs. The simulation results reported in this paper demonstrate that there is no "black and white" answer to whether having prediction is better than not having it, or whether greater sensory range is more beneficial. For some sensory ranges, nonpredictive control is better than predictive control and vice versa (Figure 2. This illustration demonstrates the utility of such analyses, and points to a need for a more systematic formal treatment of cost that will allow designers to compare divergent architectures. Without the inclusion of cost assumptions, it is impossible to evaluate multi-agent architectures based on absolute performance alone.

References

[1] David Applegate, William Cook, Sanjeeb Dash, and Andr Rohe. Solution of a min-max vehicle routing problem. *INFORMS Journal on Computing*, 14(2):132–143, 2002.

[2] Elizeth G. Araujo and Roderic A. Grupen. Learning control composition in a complex environment. In *From Animals to Animats 4: Proceedings of the Fourth International Conference on Simulation of Adaptive Behavior*, 1996.

[3] R. C. Arkin. Motor schema-based mobile robot navigation. *International Journal of Robotic Research*, 8(4):92–112, 1989.

[4] Vladimir Bugera. Properties of no-depot min-max 2-traveling-salesmen problem. In S. Butenko, R. Murphey, and P. Pardalos, editors, *Recent Developments in Cooperative Control and Optimization*. Kluwer Academic Publishers, 2003.

[5] Jose Carmena and John Hallam. Improving performance in a multi-robot task through minimal communication. In *Proceedings of the 7th Symposium on Intelligent Robotic Systems (SIRS)*, July 1999.

[6] A. Drogoul and J. Ferber. From Tom Thumb to the Dockers: Some experiments with foraging robots. In *Proceedings of the Second International Conference on Simulation of Adaptive Behavior*, 1992.

[7] Bezalel Gavish and Kizhanathan Srikanth. An optimal solution method for large-scale multiple traveling salesmen problems. *Operations Research*, 34(5):698–717, 1986.

[8] David McFarland. Towards robot cooperation. In *From Animals to Animats 3. Proc. of the Third International Conference on Simulation of Adaptive Behavior*, 1994.

[9] R. Montemanni, L.M. Gambardella, A.E. Rizzoli, and A.V. Donati. A new algorithm for a dynamic vehicle routing problem based on ant colony system. In *Second International Workshop on Freight Transportation and Logistics*, 2003.

[10] J. Pearl. A^*_ϵ—an algorithm using search effort estimates. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 4, pages 392–399, 1982.

[11] S. J. Russell and D. Subramanian. Provably bounded-optimal agents. *Journal of Artificial Intelligence Research*, 1:1–36, 1995.

[12] Matthias Scheutz and Paul Schermerhorn. Steps towards a theory of possible trajectories from reactive to deliberative control systems. In Russell Standish, editor, *Proceedings of the 8th Conference of Artificial Life*. MIT Press, 2002.

[13] Matthias Scheutz and Paul Schermerhorn. Many is more but not too many: Dimensions of cooperation of agents with and without predictive capabilities. In *Proceedings of IEEE/WIC IAT-2003*. IEEE Computer Society Press, 2003.

[14] E. Spier and D. McFarland. Possibly optimal decision making under self-sufficiency and autonomy. *Journal of Theoretical Biology*, 189:317–331, 1998.

[15] E. Stergaard, G. Sukhatme, and M. Mataric. Emergent bucket brigading - a simple mechanism for improving performance in multi-robot constrainedspace foraging tasks. In *Proceedings of the 5th International Conference on Autonomous Agents*, May 2001.

[16] Paul M. Thompson and Harilaos N Psaraftis. Cyclic transfer algorithms for multivehicle routing and scheduling problems. *Operations Research*, 41(5):935–946, 1993.

[17] H. Wang and D. Xue. An intelligent zone-based delivery scheduling approach. *Computers in Industry*, 48:109–125, 2002.