# Social Coordination without Communication in Multi-Agent Territory Exploration Tasks

Paul Schermerhorn*
Artificial Intelligence and Robotics Laboratory
Department of Computer Science and Engineering
University of Notre Dame
Notre Dame, IN 46556
pscherm1@nd.edu

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

## ABSTRACT

In the recent past, several different methods for coordinating behavior in multi-robot teams have been proposed. Common to most of them is the use of communication to coordinate behavior. For many practical applications, however, communication might not be an option (e.g., because of energy constraints of embedded platforms, limited communication range of wireless transmitters, security risks of potential interception of messages in hostile territory, etc.).

In this paper we examine alternative, low-complexity, low-cost strategies without communication for coordinated agent behavior. Specifically, we investigate the utility of a "social preference mechanism" and a "pairing mechanism" in territory exploration tasks (with many practical instantiations, e.g., two robots with different sensory capabilities investigating rock formations on a remote planet), where agents have to explore their environment to find and visit $k$ checkpoints, which only count as "visited" when $n$ agents are present at them at the same time. Both mechanisms are intended to increase the likelihood of two agents being at a same checkpoint at the same time. Experimental results indicate that pairing is the better strategy, thus raising interesting questions about tradeoffs between agent complexity and group size (e.g., whether fewer but more expensive agents that have sufficient resources to visit checkpoints individually are a better choice than more less expensive agents).

## 1. INTRODUCTION

Efficient performance of multi-agent tasks often requires some form of coordination between agents. Even in seemingly simple tasks, where agents perform autonomously without the need for help from others, agents may work at odds with one another, leading to wasted effort and reduced efficiency. More complex tasks may even *require* coordination in order to perform them. Communication is often used to facilitate coordination in multi-robot tasks, yet it may not always be feasible to include communication. Other investigations use behavior coordination among robotics agents to study the utility of communication. Arkin and Hobbs [3, 2, 10], for example, compare results on robotic *retrieval tasks* between agents with and without communication. Communication was found to be beneficial if agents communicated their states, which could cause other agents to join in the task. Werger et al. [12, 13], on the other hand, found communication to be unnecessary to achieve task formation in a system which uses behavior-based mechanisms to generate cooperative behaviors, as did Quinn et al. ([8]). While some of these results are in line with the conclusions of this paper, the tasks involved are very different from MATE-$n$ tasks.

Clearly the benefits of communication are inconclusive. Factors in determining the feasibility of communication for multi-robot exploration teams include weight (communication will require additional equipment), communication range, the cost of transmitters, and the energy required to communicate and process communicated information (i.e., communication in itself is useless without some process that utilizes the extra information). In many cases, these additional factors eliminate communication as an option for coordination, so a low-cost, low-complexity strategy is needed for coordination instead. Such a strategy would avoid the high cost of communication, but must provide sufficient coordination to allow good task performance.

In this paper, we present two coordination mechanisms, a "social preference mechanism" and a "pairing mechanism" for multi-agent coordination. We test these new mechanisms in *multi-agent territory exploration* (MATE-$n$) tasks, which require a group of agents to visit all checkpoints in the environment. In order for a checkpoint to count as "visited," $n$ agents must be present simultaneously (consider a task in which extraplanetary landers with heterogeneous equipment must explore the terrain, but may not be able, individually, to accomplish scientific goals at interesting sites). When $n > 1$, the degree of coordination required to accomplish the task efficiently is substantial. Experimental results (presented below) show that, while the social preference mechanism is able to improve performance, the pairing mechanism is the better choice for coordination in MATE-$n$ tasks.

*Student Author

This paper proceeds as follows: first MATE-$n$ is specified in detail. Then the methods used are described, including the agent model that serves as a base for the experiments, as well as the experimental setup used. Then experiments and results are presented for the base agent, social preference agents, and pairing agents.

## 2. THE MATE-$N$ TASK

The *multi-agent territory exploration* (MATE-$n$) task is inspired by biological foraging tasks, but finds many applications in other domains. Agents explore their environment, looking for marks indicating the presence of a checkpoint. When enough agents ($n$) are present at a checkpoint, its mark is removed and the agents continue the task, exploring the environment until all checkpoints have been visited by $n$ agents simultaneously (note that agents do not need to *arrive* at the checkpoint simultaneously, only to be present simultaneously). More formally:

**Definition 1:** A MATE-$n$ task $T(C_n, A, R, D)$ requires a group of agents $A$, each with sensory range $R$, to visit a set of checkpoints $C$, each of which requires $n$ agents present to remove its mark, in a 2D environment where agents and checkpoints are placed according to a probability distribution $D$.

Performance is defined as the time it takes to remove all checkpoint marks from the environment.[1] Thus, unlike many natural foraging tasks, agents in MATE-$n$ tasks are working as a group. Individual agents' performances are important insofar as they contribute to the group's performance, but individuals are not rewarded for good performance.

Typically, agents do not have *a priori* knowledge of the distribution $D$; both agents and checkpoints are distributed randomly in the environment, requiring agents to explore to locate checkpoints. If this information *is* available, it is possible to calculate an optimal offline solution by exhaustively searching the entire space of checkpoints assigned to each agent, along with the order in which each agent visits its assigned checkpoints. Such a solution, however, is computationally intractable.

MATE-$n$ tasks are similar in some ways to Traveling Salesman Problems (TSP) [5], and even more so to multi-Traveling Salesman Problems (MTSP), in which some member of a sales team is required to visit each city. The delivery scheduling problem is an instance of MTSP for which approximations exist [6, 4, 11, 1, 7, 9]. However, because TSP variants require agents to return to their initial positions, no TSP approximation fits exactly the MATE-$n$ task; the optimal visit order is affected by requiring a round trip.

## 3. METHODS

We begin by investigating the performance of a base agent model in a MATE-$n$ task, then add a new (coordinating) feature $F$ to the architecture and compare the performance of the new agent kind with the performance of the old. The remainder of this section describes the base agent model used to obtain absolute performance, and the experimental setup employed.

[1]Other measures of performance could be the total distance traveled by all agents in $A$, or the difference in length between all agents' paths (as in a load balancing goal)

### 3.1 Agent Model

The simplified sensor model employed by the agents ignores sensor modality, assuming that agents can sense all checkpoints in a full 360° radius within a sensory range of $R$. The agent's sensory input $S$ is combined with its inner states $I$ to produce behaviors according to the agent function $F$. $F$ is a mapping $F : 2^S \times I \rightarrow D$, where $2^S$ is the power set of sensory data given as pairs $< d, \alpha >$ of distances $d \in [0, R]$ and angles $\alpha \in [0, 2\pi]$, $I$ is a set of inner states (possibly empty), and $D \in [0, 2\pi]$ is the set of possible directions the agent will move in.

The base agent model represents the best-performing asocial architecture from previous work: the *timeout* agent. For MATE-1 tasks, a simple agent that targets for collection the closest checkpoint detected is sufficient for reasonably good performance. When $n > 1$, however, these simple agents are susceptible to 'deadlock' states in which each agent moves to the nearest checkpoint and waits for another agent to arrive; when all agents have found (distinct) checkpoints, the task cannot proceed. Timeout agents incorporate a *wait timer* $w$ that is started when the agent arrives at a checkpoint and expires after a fixed duration $W$. At that point, the checkpoint is added to the agent's *filter set*, which forces the agent to ignore the checkpoint. Effectively, the agent does not perceive a checkpoint that is in its filter set. After a fixed cycle count, the checkpoint is removed from the filter set, and the agent can once again perceive the checkpoint (if it is within sensory range).

The following ruleset comprises the agent model for timeout agents:

**Rule 1:** if no checkpoint is perceived ($S = \emptyset$), perform a random walk $RW(rwd, \beta)$ (i.e., move in the direction of the current heading $\theta$ for $rwd$ cycles, then change heading randomly to some value in $[\theta - \beta, \theta + \beta]$)

**Rule 2:** if some checkpoint $C$ is within visiting distance and there are $n$ agents within visiting distance of $C$, remove the checkpoint's mark and reset $w$ to 0

**Rule 3:** if some checkpoint $C$ is within visiting distance, there are fewer than $n$ agents at $C$, and $w = W$, reset $w$ to 0 and add $C$ to the agent's filter set

**Rule 4:** if some checkpoint $C$ is within visiting distance, there are fewer than $n$ agents at $C$, and $w < W$, increment $w$

**Rule 5:** if no checkpoint is within visiting range but some checkpoints are perceived ($S \neq \emptyset$), go directly towards the closest checkpoint (the direction is given by $\alpha$ such that $\min_d [\langle d, \alpha \rangle | \langle d, \alpha \rangle \in S]$)

These simple agents employ a random walk mechanism that changes their heading after $rwd$ cycles without perceiving a checkpoint. The value of $rwd$ was determined for each combination of group size and sensory range employed by systematically exploring the space of values and selecting those with the best performance. Additionally, the environment is bounded; agents "bounce" off of the edge of the environment with a small amount of random error. Agents will move directly to the nearest perceived checkpoint (when there is one) and wait there for a total of $n$ agents to arrive. After waiting for $W$ cycles, the checkpoint is added to the agent's filter set and the agent can then target other perceived checkpoints or begin foraging via the random walk mechanism. The value of $W$ was determined similarly to

*rwd*: a systematic exploration of the performance space of each group size/sensory range combination yielded the value of $W$ with the best performance; this is the value of $W$ for that combination. A counter is associated with each member of the filter set that is incremented at every cycle. When the counter reaches $f$, the checkpoint is removed. The value of $f$ is fixed at 200.

## 3.2 Experimental Setup

To begin exploring coordination in MATE-$n$ we must choose a reasonable value for $n$. While there is some degree of coordination required when $n = 1$, for the most part it comes down to keeping out of each other's way; cooperation is not required in visiting individual checkpoints in MATE-1. MATE-2 provides a task in which agents must actively coordinate checkpoint visits if efficient performance is expected. The chances of two agents being at the same checkpoint simultaneously are small if there is no coordination effort. The challenge is similar in kind for $n = 3$, $n = 4$, etc., albeit with increasing difficulty. Hence, for this paper we select $n = 2$ for the number of agents that must be present at a checkpoint.

Each experiment reported in this paper consists of 40 experimental runs using different randomly generated initial conditions in a continuous 2D world, which is limited to an 800 by 800 square region. Sensory range is varied from 25 to 250 in steps of 25, and from 300 to 800 in steps of 50. Group size is varied from 2 (the minimum size required to successfully complete the MATE-2 task) to 10. The same set of 40 initial placements of checkpoints ($|C| = 10$) is used for all experiments. Moreover, the same 40 initial placements of agents is used for all experiments with identical group size (i.e., all experiments with $|A| = 2$ will use the same 40 initial conditions, regardless of sensory range or agent type, as will all experiments with $|A| = 3$, etc.). 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. Agents are allowed 10,000 cycles within which to complete the task; after that point, the experimental run is considered a failure, and its value is set at 10,000. Thus, 10,000 cycles is the ceiling for performance.

# 4. TIMEOUT AGENTS

The first set of experiments measures the performance of the base architecture, the timeout agent. Recall that this agent forages for checkpoints in the environment and goes to the nearest one it sees. If there is another agent at the checkpoint, the mark is removed and the agents move on. Otherwise, the agent waits at the checkpoint for $W$ cycles, or until another agent arrives and the mark can be removed.

## 4.1 Results

The performance results for timeout agents are presented in Figure 1 (left). Overall, the trends are predictable: as sensory range increases, performance increases (i.e., average cycles to completion decreases), to a point. When sensory range is greater than 300, performance levels off, and there is little benefit to increased range. Similarly, performance increases with group size, but improvements diminish as size increases so that, for example, there is little difference in performance between $|A| = 9$ and $|A| = 10$.

A two-way 9x21 ANOVA with *group size* (2-10) and *sensory range* (25-800) as randomized variables and *average cycles to completion* as the dependent variable shows highly significant main effects for group size ($F(8, 7371) = 1983.8795, p < 0.001$) and range ($F(20, 7371) = 207.4370, p < 0.001$), confirming that increasing group size and sensory range improves performance. There is also a highly significant interaction between group size and sensory range ($F(160, 7371) = 5.8574, p < 0.001$); this confirms that larger groups increase performance faster as sensory range increases, reaching earlier the point at which added sensory range is of no benefit. These results are exactly what one would expect; essentially we are increasing the resources available, and should, therefore, expect performance to increase. Similarly, we should expect that there would come a point at which more resources will be of little benefit. These are independent of the type of agent tested, so this pattern will be repeated in subsequent results. Therefore, while we will continue to report the values for the randomized variables group size and sensory range, we will not discuss them specifically in subsequent results, except when they interact significantly with other variables tested.

## 4.2 Analysis

As expected, two trends are evident in the results: larger groups outperform smaller groups, and higher sensory ranges outperform smaller sensory ranges. Several additional functional roles are displayed, each of which is a contributor to poor performance:

**Jilted Groom:** an agent at a checkpoint whose wait timer expires (the bride never arrives)

**Jilted Bride:** an agent that targets an occupied checkpoint, but the occupying agent's wait timer expires before the "bride" arrives

**Lonely Bachelor:** an agent that spends a prolonged period searching for a checkpoint

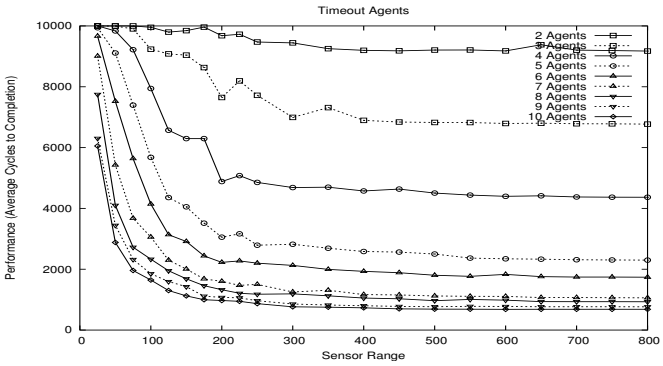
**Unlucky Bachelor:** an agent caught in a cycle of waiting for the full length of the wait timer and then foraging again for another checkpoint, only to have its wait timer expire again

**Dedicated Bachelor:** similar to the unlucky bachelor, except that there are other occupied checkpoints within sensory range, but the agent continues to target unoccupied checkpoints (seemingly preferring to remain a bachelor)

**Bridesmaid:** an agent that targets a checkpoint that is already targeted by two nearer agents, and so is destined to arrive late, after the checkpoint has been visited ("always a bridesmaid, never a bride")

With low sensory range, agents tend to spend a lot of time as lonely bachelors because of the difficulty of locating a checkpoint. Moreover, once they do locate a checkpoint, they tend to play the part of the jilted groom, because the other agents have the same difficulty locating the occupied checkpoint due to their own low sensory ranges. The wait timer expires and the groom moves on, returning to the bachelor role. As sensory range increases it becomes easier to locate checkpoints, and the lonely bachelor is seen less often, while the frequency of the groom role increases.

When the number of agents is small, agents have similar difficulties: when they do locate a checkpoint, there is often no other agent in a position to play the role of bride. Instead they tend to assume the unlucky bachelor role, a prolonged



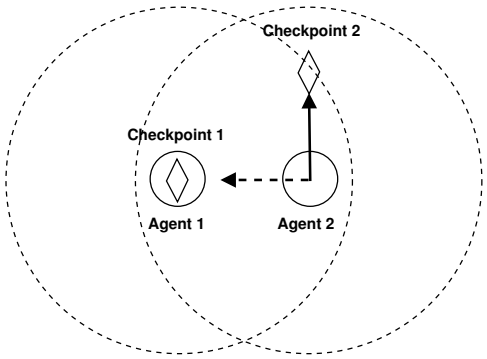




**Figure 1: Left: Average performance of timeout agents for two to ten agents and sensory ranges from 25 to 800. Right: A difficult situation for timeout agents to handle: the default behavior of Agent 2 is to move to Checkpoint 2, as a result of Rule 2. However, with Agent 1 at Checkpoint 1, a better strategy would be to move to Checkpoint 1.**

cycle of jilted groom episodes. This is because they are unlikely to be placed near each other in the environment, so there may be a long distance between them that a potential bride will have to traverse. Note that increasing sensory range does not address this problem. Timeout agents target the closest checkpoint detected, and there will often be checkpoints between them, preventing potential brides from arriving before the wait timer expires. Fortunate initial placement can mitigate this problem, and agents may drift toward one another, allowing them to eventually break out of the unlucky bachelor role. However, in the case of 2-agent groups, this seldom happens.

The simulations are limited to 10,000 cycles, as described above. The results for 2 agents in Figure 1 (left) show that they do not even approach 9,000 cycles. This is because they fail to complete in most cases, regardless of the sensory range. 3- and 4-agent groups improve, but still fail in many experimental runs. 5 agents appear to be sufficient to overcome the unlucky bachelor problem.

## 4.3 Discussion

The timeout mechanism was designed to break deadlocks by limiting the amount of time agents spend waiting at checkpoints for other agents to arrive. The mechanism is not perfect (e.g., there are times when an agent's wait timer will expire just before another agent arrives, giving rise to the jilted bride), but overall performance is improved. For timeout agents, some wait cycles are inevitable; the first agent arriving at a checkpoint must wait for another agent to arrive (unless we expect them to arrive simultaneously–highly unlikely!). However, some wait cycles are especially wasteful.

Figure 1 (right) presents a scenario in which Agent 1 is waiting at Checkpoint 1 but Agent 2 is closer to Checkpoint 2. In this situation, Agent 2 will move to Checkpoint 2 and wait $W$ cycles. This is less efficient than moving to Checkpoint 1 in most cases, because Agent 1 will wait until its timer expires to leave Checkpoint 1. Furthermore, it is possible that Checkpoint 2 will be in Agent 1's filter set, in which case it will not move there to join Agent 2. Finally, it is possible (and, in our observations, frequently the case) that Agent 2's timer will expire before Agent 1 arrives, in which case they may enter a cycle of flipping between checkpoints without marking any visited. These are examples of the dedicated bachelor functional role, in which Agent 2 appears to intentionally avoid targeting a checkpoint with a waiting groom, although this behavior is really a result of the same control architecture that would require Agent 2 to move to Checkpoint 1 if it were nearer.

As a measure of how often agents make the dedicated bachelor mistake, we tracked *redundant wait cycles*. A redundant wait cycle is one spent by an agent waiting at a checkpoint while another agent is waiting at another checkpoint in sensory range. The average number of redundant wait cycles for all agents in a simulation is the measure of *Redundant Wait Cycles* ($RWC$) for the simulation:

$$RWC = \frac{\sum_{n=1}^{|A|} RedundantWaitCycles(a[n])}{|A|} \quad (1)$$

Figure 2 (upper left) shows the proportion of the total cycles in Figure 1 that timeout agents spent redundantly waiting. The proportion is substantial, particularly for smaller groups with higher sensory range. Agents in larger groups are not as susceptible to redundant wait cycles because there is more frequently another agent near enough to target the checkpoint at which an agent is waiting. Similarly, when sensory range is low, agents are not as likely to perceive other checkpoints, including other checkpoints with agents waiting. A direct coordination mechanism that targets redundant wait cycles for reduction should be particularly useful for increasing performance in MATE-2 tasks, particularly in small groups with high sensory range.

# 5. SOCIAL PREFERENCE AGENTS

One simple, and seemingly natural, architectural enhancement is to modify the timeout architecture so that the agent takes into account states of other agents, preferring occupied checkpoints over checkpoints with no agent. The agent model is modified by changing Rule 5:

**Rule 5a:** if no checkpoint is within visiting range, at least one checkpoint is perceived ($S \neq \emptyset$), and there is another agent waiting at a perceived checkpoint, go directly towards the closest checkpoint with

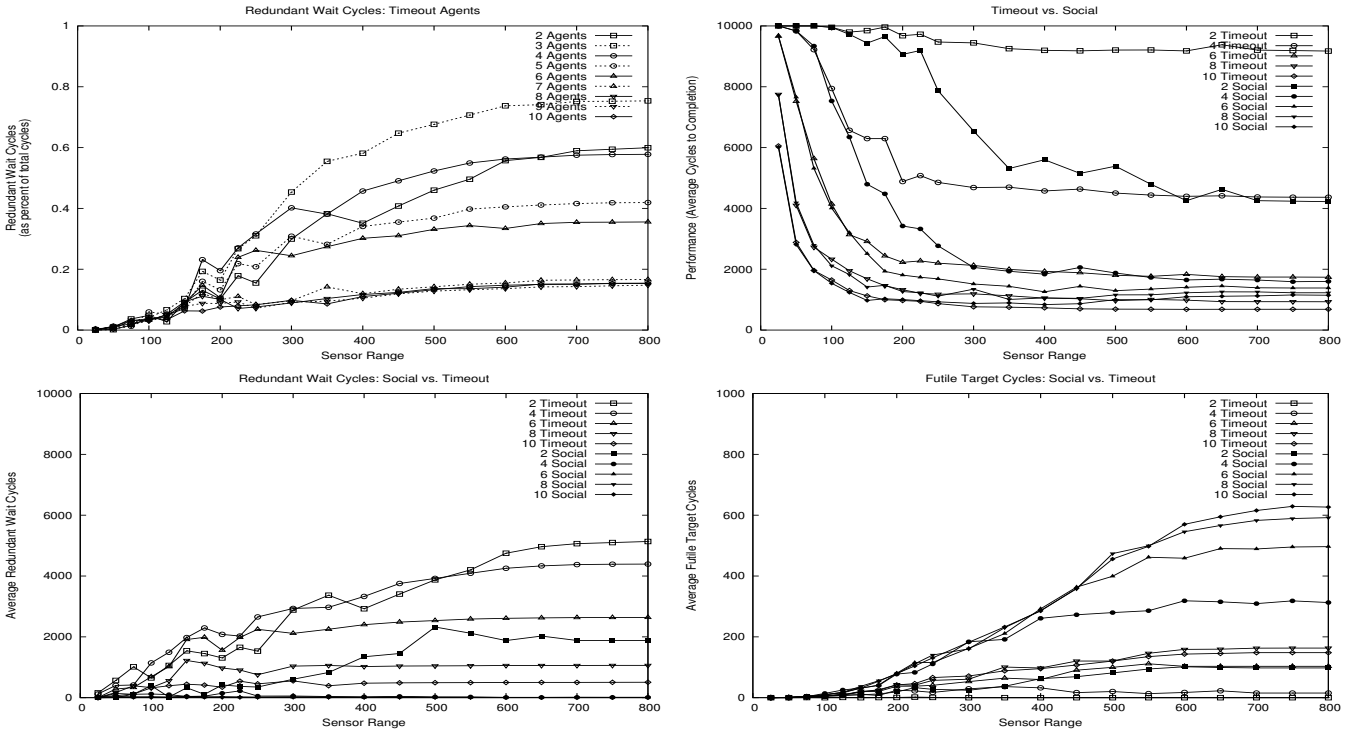



**Figure 2: Upper left: Portion of total cycles spent in redundant wait. Upper right: Average performance of timeout agents and social agents. Lower left: Comparison of redundant wait cycles for timeout agents and social preference agents. Lower right: Futile target cycles for target-closest agents and social preference agents.**

an agent (the direction is given by $\alpha$ such that $\min_d [\langle d, \alpha \rangle | \langle d, \alpha \rangle \in S]$)

**Rule 5b:** if no checkpoint is within visiting range, at least one checkpoint is perceived ($S \neq \emptyset$), and there is no other agent waiting at a perceived checkpoint, go directly towards the closest checkpoint (the direction is given by $\alpha$ such that $\min_d [\langle d, \alpha \rangle | \langle d, \alpha \rangle \in S]$)

This modified ruleset constitutes the agent model for *social preference* agents. Social preference agents will ignore nearer checkpoints when there is an agent at another checkpoint within sensory range. This modification addresses the inefficiency in Figure 1 (right); social preference agents avoid the dedicated bachelor role in most cases, although they are more susceptible to becoming bridesmaids.

## 5.1 Results

Social preference agents prefer as targets checkpoints with another agent present. This simple modification leads to dramatically increased performance–in most cases. Figure 2 (upper right) is a comparison of timeout agents and social agents. In virtually all cases, social agents outperform timeout agents. A three-way 9x21x2 ANOVA for *group size* (2-10), *sensory range* (25-800), and *agent kind* (timeout and social preference) as randomized variables with the dependent variable *average cycles to completion* finds highly significant main effects for group size ($F(8, 14742) = 3433.6255, p < 0.001$), sensory range ($F(20, 14742) = 741.4416, p < 0.001$), and agent kind ($F(1, 14742) = 998.0511, p < 0.001$). This confirms that there is an absolute performance difference between groups. There are highly significant two-way interactions between group size and sensory range ($F(160, 14742) = 20.6668, p < 0.001$), between group size and agent kind ($F(8, 14742) = 211.8039, p < 0.001$), and between range and agent kind ($F(20, 14742) = 20.1746, p < 0.001$), and a highly significant three-way interaction between all three factors ($F(160, 14742) = 6.7453, p < 0.001$). The interactions confirm the prediction (based on Figure 2, upper left) that small groups with high sensory range would benefit most from the social preference mechanism. Interestingly, the performance of large groups with high sensory range is actually *harmed* by the social preference modification.

## 5.2 Analysis

Social preference agents outperform timeout agents overall, with the interaction between group size and agent kind confirming that the mechanism is more effective in small groups, and the interaction between sensory range and agent kind confirming that it is most effective for higher sensory ranges. Performance is nearly identical at small sensory ranges for all group sizes, because in those cases there is rarely an opportunity for Rule 5a to fire; few checkpoints are detected, with or without agents. As sensory range increases, social preference agents soon become better.

The performance advantage of social preference agents is explained by the coordinating effects of the preference modification. Figure 2 (lower left) compares redundant wait cycles of social preference and timeout agents. In most cases, social agents experience fewer.

It may seem surprising that social preference agents ex-

perience *any* redundant wait cycles. If they ignore closer checkpoints when they perceive another agent at a checkpoint, the only situation in which two agents should be at different checkpoints within sensory range is when they arrive at their checkpoints simultaneously. This would certainly be the case if agents perceived all checkpoints within sensory range. However, there is an interaction here between the timeout mechanism's filter set and the social preference mechanism. Consider a situation in which agent $A_1$ is at checkpoint $C_1$ and agent $A_2$ comes within sensory range. $A_2$ will move toward $C_1$, whether there are closer checkpoints or not. In the meantime, $A_1$'s wait timer expires. $A_1$ leaves $C_1$, placing it in its filter set. While $A_1$ is traveling to $C_2$, $A_2$ arrives at $C_1$, but $A_1$ does not perceive $C_1$, so it does not turn around. When $A_1$ arrives at $C_2$, redundant wait cycles begin to accrue.

## 5.3 Discussion

Social agents see little benefit when sensory range is low, because agents are less likely to see two checkpoints at the same time and, therefore, less likely to be in a situation where their added capability is beneficial. When sensory range is high, agents have more information at any given time to which they can apply the preference. This suggests that the primary obstacle to agent performance is acquiring information; coordination is of little use in the absence of sufficient information about the environment. Once the information hurdle has been overcome, however, the benefit of coordination can be seen in small groups.

Similarly, there is little benefit when group size is high, because there are enough agents in the environment to minimize the need for coordination; agents are not as likely to be penalized by "wrong" decisions, because there is likely to be another agent nearby to visit the occupied checkpoint. Increasing group size is one way to increase the probability that two agents will target the same checkpoint. The value of $|A|$ at which increasing group size is no longer beneficial will depend on the size of the environment and the number of checkpoints that must be visited. In the environments studied, the benefit of introducing the social preference mechanism into groups of eight or more agents is probably not worthwhile. However, for $|A| < 8$, social agents significantly outperform their communicating timeout peers. When group size is small, it will often be the case that there is only one additional agent within sensory range of an occupied checkpoint. In such cases it is important for the second agent to make a good decision, because it is unlikely that another agent will happen along.

Social preference agents are more susceptible to becoming "bridesmaids" because, while they do take the state of the groom into account, they pay no attention to other potential brides. Hence, they spend time futilely targeting a checkpoint when they have no chance of being the bride. A cycle is considered *futile* if the agent targets a checkpoint but subsequently fails to visit it (i.e., because another pair of agents arrives first). More formally, the measure *Futile Target Cycles* ($FTC_A^{cyc}$) for a simulation run is the average number of futile target cycles for all agents in $A$ in the simulation run (of *cyc* cycles):

$$FTC_A^{cyc} = \frac{\sum_{a \in A} FutileTargetCycles^{cyc}(a)}{|A|} \quad (2)$$

where $FutileTargetCycles^{cyc}(a)$ is the total number of futile target cycles of agent $a$ in the given simulation run of *cyc* cycles. The average of $FTC_A^{cyc}$ for all simulation runs can then be taken as a measure of how susceptible the agents were to wasting cycles targeting ultimately unvisitable checkpoints.

As noted, social preference agent performance becomes worse than normal timeout agent performance in large groups with high sensory range. As Figure 2 (lower right) indicates, this is due to an increase in futile target cycles. When all social preference agents perceive most or all checkpoints in the environment, any time an agent arrives at a checkpoint, every agent will target that checkpoint, even when it is remote and the agent has no chance of arriving before another agent. This distracts agents from foraging locally, requiring them to return in order to visit checkpoints left behind. Note that, while the difference in scale between the graphs for redundant wait cycles and futile target cycles makes then look similar, the impact of futile target cycles is a much smaller factor in overall performance. However, when resources are plentiful (i.e., large groups with high sensory ranges), the social preference mechanism harms performance even without taking its added cost into account. This inefficiency could be dealt with, at least in part, with the addition of mechanisms that observe and use the states of other brides when making control decisions. However, we decided to try a different approach, presented in the next section.

# 6. PAIRING AGENTS

As we mentioned before, MATE-$n$ tasks for $n > 1$ require greater degrees of coordination than MATE-1 tasks, because when $n = 1$, there is no need to coordinate actual checkpoint visits. Any time an agent arrives at a checkpoint, it is visited. This suggests a different approach to improving performance on MATE-2: rather than adding increasingly more complex mechanisms to attempt to deal with coordination inefficiencies, we can design agents that change the character of the task itself, superimposing a different, simpler task on the original one. Specifically, MATE-2 can be reduced to MATE-1 by simply allowing the agents to "team up" and proceed in pairs. After the initial pairing off phase, agent pairs should be perfectly coordinated: there will never be a wait cycle if both agents always arrive simultaneously. We created *pairing* agents that do this, to see how effective explicitly teaming up can be. The following ruleset constitutes the agent model for pairing agents:

**Rule S1:** if $role = scout$ and no other agent is perceived, perform a random walk $RW(rwd, \beta)$

**Rule S2:** if $role = scout$, some other agent $A$ is perceived, $A$'s $role = scout$, and $A$'s leadership characteristic $L$ is greater than own, $role = follower$ and $leader = other$

**Rule S3:** if $role = scout$, some other agent $A$ is perceived, $A$'s $role = scout$, and own leadership characteristic $L$ is greater than $A$'s, $role = leader$

**Rule F1:** if $role = follower$, follow the leader

**Rules L1–L5:** if $role = leader$, proceed with rules 1–5 for timeout agents, given above, with $W = 10000$

A pairing agent can be in one of three states: scout, leader, and follower. The scouting phase is when the agent is looking for a partner. During this phase, the agent disregards

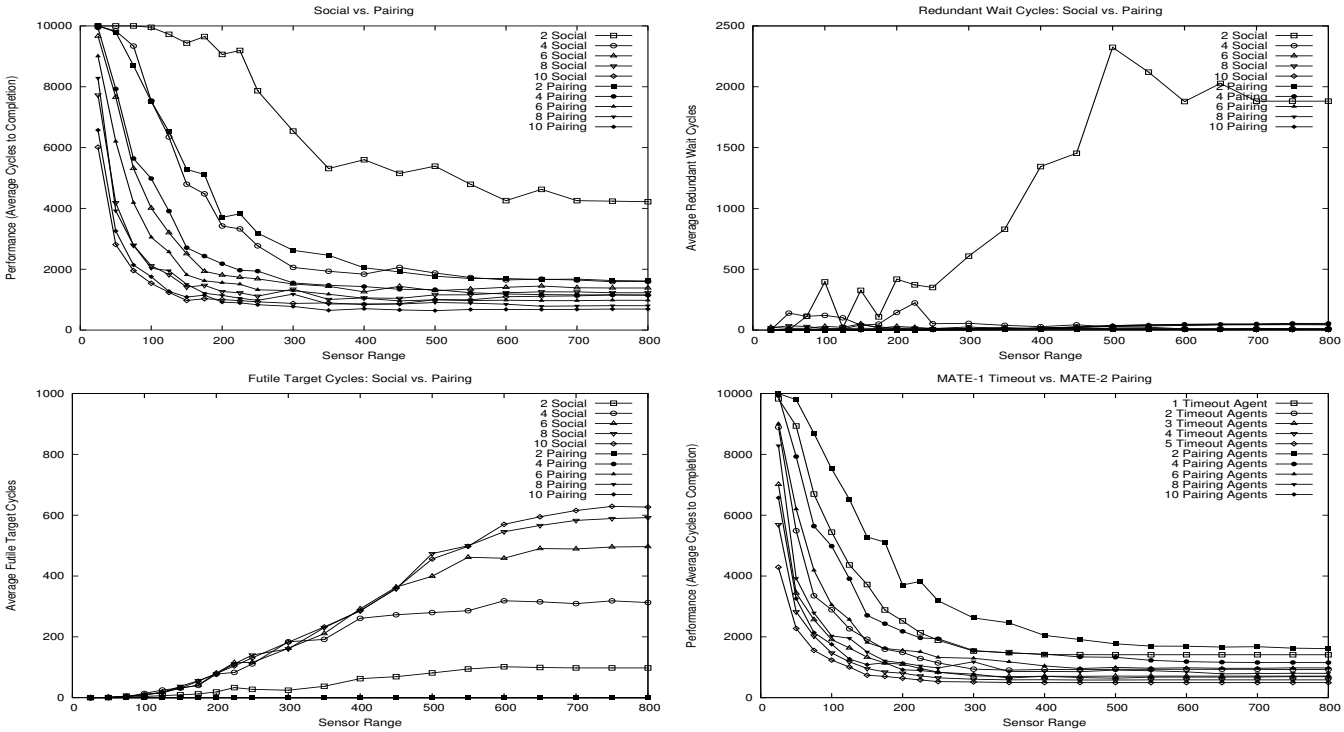



**Figure 3: Upper left: Average performance of pairing and social agents. Upper right: Redundant wait cycles for pairing agents and social preference agents. Lower left: Futile target cycles for pairing agents and social preference agents. Lower right: Average performance of 1 to 5 timeout agents in MATE-1 and 2 to 10 pairing agents in MATE-2.**

checkpoints, searching only for other agents. When another (scout) agent is located, they compare their leadership characteristic to determine who will be the leader. Whichever agent's leadership quotient is higher leads, the other follows. If more than two agents are within sensory range, the top two pair off, then the next two, etc. Any odd agent is left out and continues to scout for a partner.

Once agents have formed a team, one enters the leading phase, while the other enters the following phase. In the leading phase, the agent effectively becomes a timeout agent. It explores the environment looking for checkpoints and moving to them when detected. In the following phase, the agent ignores all checkpoints, blindly following the leader wherever it decides to go.

## 6.1 Results

Figure 3 (upper left) compares pairing agents with non-communicating social agents. The performance improvements in small groups is quite good, although large groups of social agents perform as well as or better than pairing agents at low sensory range. A three-way 9x21x1 ANOVA was conducted with *group size* (2-10), *sensory range* (25-800), and *agent kind* (social and timeout) as the randomized variables and *cycles to completion* as the dependent variable. Group size ($F(8, 14742) = 2357.9752, p < 0.001$), sensory range ($F(20, 14742) = 2039.0519, p < 0.001$), and agent kind ($F(1, 14742) = 1692.0771, p < 0.001$) were all found to be highly significant main effects for cycles to completion, confirming that, overall, pairing agents outperform social preference agents. There is a highly significant two-way interaction between group size and agent kind ($F(160, 14742) = 47.7577, p < 0.001$), confirming that increases in sensory range are most effective in smaller groups, another highly significant two-way interaction between group size and agent kind ($F(8, 14742) = 276.9290, p < 0.001$), confirming that the greatest performance differences between social and pairing agents are to be found in smaller groups, and a final highly significant two-way interaction between sensory range and agent kind ($F(20, 14742) = 13.9703, p < 0.001$), confirming that the performance differences between social and pairing agents varies with sensory range. Finally, a highly significant three-way interaction ($F(160, 14742) = 8.1646, p < 0.001$) is a result of the social agents having greater performance in large groups with low sensory range, but worse performance in the remaining configurations.

## 6.2 Analysis

Pairing can almost be viewed as the ultimate in coordination for tasks like MATE-$n$. Once a pair has teamed up, the effect of the above coordination issues is minimal. Figures 3 (upper right) and 3 (lower left) confirm this. These demonstrate that the two sources of inefficiency are virtually eliminated by the pairing mechanism, thereby leading to improved performance. Note that they are not eliminated completely; it is still possible for two leaders to target the same checkpoint, leading to futile target cycles for the second pair. Similarly, leaders and followers do not always arrive at a checkpoint simultaneously,[2] so it is possible, particularly with high sensory range, for two leaders within sen-

[2] They can arrive separately at the first checkpoint they

sory range to be waiting for their followers to arrive. Note, however, that these redundant wait cycles do not have any negative impact on performance.

# 7. DISCUSSION

The experimental results presented here show that the pairing mechanism is a good solution for coordination in MATE-$n$, without the need for communication. The degree of coordination achieved is high, after the initial pairing phase. The jilted groom role is eliminated, because agents do not target checkpoints until they have paired with another agent, which will arrive simultaneously, except possibly the first checkpoint visited (see Footnote 2). Similarly, the leader will not leave the checkpoint before the follower arrives, eliminating the jilted bride role. The lonely bachelor can still be a problem, in both the scout and leader/follower phases, when sensory range is low, but this is unavoidable and not a coordination issue. There are no dedicated bachelors, because, again, agents do not begin targeting checkpoints until they are paired with another agent. The final detrimental functional role, the bridesmaid, is the only one that still negatively affects pairing agents, in this case when one *pair* of agents arrives at a checkpoint before another pair. However, the magnitude of its impact is significantly decreased (especially at high sensory ranges) relative to social preference agents, which are particularly susceptible to playing the bridesmaid under these circumstances.

The bottom right graph in Figure 3 compares the performance of pairing agents in MATE-2 with the performance of timeout agents in MATE-1, for the same environmental conditions. The MATE-1 results are for 1–5 agents, whereas the MATE-2 results are for 2, 4, 6, 8, and 10 agents. If we compare the performance of a particular group size $A$ in MATE-1 with the performance of $2 \cdot A$-agent groups in MATE-2, we can get a close approximation of the initial cost of teaming up for pairing agents. The figure shows that, while performance of pairing agents in MATE-2 is similar to the performance of timeout agents in MATE-1, there is a substantial penalty for the initial teaming up period. This suggests that, if it were possible to construct single agents that are capable of performing the task at a checkpoint individually at a reasonable cost, it would be beneficial to do so (thereby completing the reduction of the task to MATE-1). What is a "reasonable cost?" If the new agent kind were to cost less than or equal to twice what the old agent kind cost, this would clearly be a win. Moreover, given the cost of pairing shown in Figure 3, the cost of the enhanced agent could actually be somewhat more than twice that of the original MATE-2 pairing agents, and still achieve a better performance-cost ratio than the pairing agents.

# 8. CONCLUSION

In this paper we demonstrated that a simple mechanism like pairing can give rise to very high performance in MATE-$n$ tasks. This is not only a good alternative to solutions with communication in situations in which communication is either not possible or is too expensive, but may actually be the better solution overall. Degenerate forms of leadership strategies like this one may in general provide good alternatives to communication for even more complex tasks than MATE-$n$. Future work will investigate this hypothesis for the $\bigcup_{k=1}^{n}$ MATE$-k$ tasks, in which checkpoints require from 1 to $n$ agents, and there may be no way to predict the value of $k$ for any particular checkpoint without being within sensory range of the checkpoint, where agents can perceive what kind of checkpoint it is (i.e., where they can figure out $k$).

visit, because when they initially pair up, they are at a distance $R$ from each other.

# 9. REFERENCES


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

[2] R. C. Arkin. Cooperation without communication: Multiagent schema-based robot navigation. *Journal of Robotic Systems*, 9(3):351–364, 1992.

[3] R. C. Arkin and J. D. Hobbs. Dimensions of communication and social organization in multi-agent robotic systems. In *Proceedings of the Second International Conference on Simulation of Adaptive Behavior*, pages 486–493, 1992.

[4] V. 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] T. T. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms*. MIT Press, Cambridge, MA, 1990.

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

[7] R. Montemanni, L. Gambardella, A. Rizzoli, and A. 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.

[8] M. Quinn, L. Smith, G. Mayley, and P. Husbands. Evolving controllers for a homogeneous system of physical robots: Structured cooperation with minimal sensors. *Philosophical Transactions of the Royal Society of London, Series A: Mathematical, Physical and Engineering Sciences*, 361:2321–2344, 2003.

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

[10] A. Wagner and R. Arkin. Multi-robot communication-sensitive reconnaissance. In *Proceedings of the 2004 IEEE International Conference on Robotics and Automation*, 2004.

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

[12] B. B. Werger. Cooperation without deliberation: a minimal behavior-based approach to multi-robot teams. *Artificial Intelligence*, 110(2):293–320, 1999.

[13] B. B. Werger and M. J. Mataric. From insect to internet: Situated control for networked robot teams. *Annals of Mathematics and Artificial Intelligence*, 31(1–4):173–198, 2001.