# Design and Evaluation of Distributed Role Allocation Algorithms in Open Environments

Myriam Abramson
Naval Research Laboratory
Myriam.Abramson@nrl.navy.mil
Fax: 202-404-1191

William Chao
Naval Research Laboratory
chao@itd.nrl.navy.mil
Fax: 202-767-1191

Ranjeev Mittu
Naval Research Laboratory
Ranjeev.Mittu@nrl.navy.mil
Fax: 202-404-1191

***Abstract*— Role allocation has emerged as one of the key issue in teamwork involving communication. While direct point-to-point communication is expensive and uncertain, access to neighbors is reliable and efficient in scale-free networks such as those found in open environments and social networks. This suggests decentralized approaches to role allocation based on communication between neighbors. This paper adapts and evaluates some of the basic types of algorithms in distributed role allocation in open environments using a novel coordination measure in the prey/predator domain.**

***Index Terms*— distributed problem-solving, role allocation, coordination, open environments**

## I. Introduction

Open environments, e.g. peer-to-peer (P2P) and wireless or Mobile Adhoc Networks (MANET), are characterized by dynamic team formation, multiple and changing goals, non-deterministic state transitions, variable resources, and unreliable communication. The key issue in teamwork theory is role allocation based on the communication of beliefs and intentions. This issue becomes complicated by the characteristic factors of open environments.

Role allocation can be viewed as a constraint satisfaction problem similar to resource allocation problems where agents are variables and roles are values that those variables can take. When a utility or preference is attached to an assignment, role allocation also entails a constraint optimization problem. When multiple teams coexist, an agent can take on multiple non-conflicting roles. Over-constrained situations where a team cannot be formed because of an insufficient number of agents are not relevant here. Rather, an agent has to assume multiple roles in those situations. The capability of agents to handle multiple roles is an important aspect of the coordination of intelligent agents.

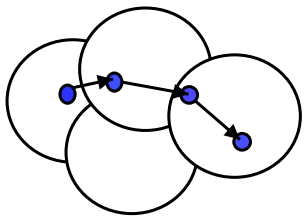

Fig. 1. Multi-hop routing in MANET
Communication can extend beyond the immediate neighbors, effectively extending the communication range of the agents

Mobile Adhoc Networks are characterized by a dynamic topology due to node mobility and limited energy life. In addition, the limited communication range provides only a partial knowledge of the global environment but is not necessarily restricted to the immediate neighbors (see Fig. 1). Those constraints make it advantageous for agents to self-organize within their communication range and the absence of centralized control requires a distributed control policy. The uncertainty that a message will arrive at its destination in a finite amount of time violates one of the basic communication assumptions of distributed constraint satisfaction algorithms[1]. How to extend those algorithms to open and uncertain environments is still an active area of research[2].

This paper is organized as follows. Section II introduces our methodology for evaluation as well as motivates experiments in the prey/predator domain. Section III presents results and analysis of our evaluation. Finally, Section IV and V conclude with a summary of related work and recommendations

for future work.

## II. Methodology

Our methodology consists of adapting three distinct types of distributed role allocation algorithms and evaluating results in the prey/predator domain – a concurrent and asynchronous environment.

### A. Prey/Predator Problem

The prey/predator pursuit game (Fig. 2) is a canonical example in the teamwork literature[3] because one individual predator alone cannot accomplish the task of capturing a prey. Practical applications of the prey/predator pursuit game include, for example, unmanned ground/air vehicles target acquisition, distributed sensor networks for situation awareness, and rescue operations. Due to the decomposability of the global reward as a sum of local rewards, the original problem can be extended to multiple teams by including additional preys. Prey/predators can sense each other if they are in proximity but do not otherwise communicate. Predators communicate with other predators individually or can broadcast messages through their neighbors. Four predators are needed to capture a prey by filling out four different roles: surround the prey to the north, south, east and west. Those roles are independent of each other and can be started at any time obviating the need for scheduling. The only requirement is that they have to terminate at the same time either successfully when a capture occurs or unsuccessfully if no team can be formed. The predator agents are homogeneous and can assume any role but heterogeneity can be introduced by restricting the role(s) an agent can assume. The prey and predators move concurrently and asynchronously at different time steps. In addition to the four orthogonal navigational steps, the agents can opt to stay in place. In case of collision, the agents are held back to their previous position. The prey moves furthest away from the closest predator in its perception range or does not move if no predators are in sight. This strategy is referred to as the Move Away From Nearest Predator (MAFNP) strategy [4]. It has been shown to lead to certain capture situations[5]. The preference $u_{ij}$ of predator agent $i$ for a role $j$ is inversely proportional to the Manhattan distance $d$ required to achieve the role.

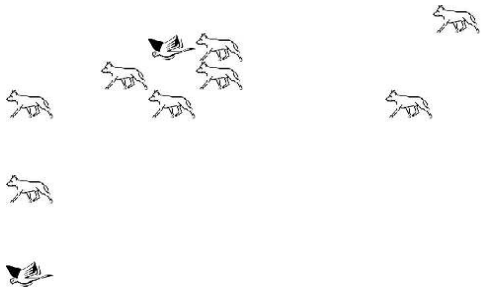

Fig. 2. Prey/predator pursuit game on a toroidal grid

The predators move in the direction of their target when assigned a role or explore the space according to a memory-based scheme on the last few steps. The decision space for the role allocation of $P$ predators and $p$ preys is $O(p^T)$ where $T$ is the number of teams of size $t$[1]. This problem belongs to the most difficult class of problems for constraint satisfaction in multi-agent systems[6] due to the dynamic nature of the environment and the mutually-exclusive property of role allocation.

### B. Distributed Role Allocation Algorithms

There are three basic types of distributed role allocation algorithms involving communication between agents. Typically, each agent runs the same algorithm to decide whether to change their value and arrive at a solution after exchanging messages in an iterative fashion. In one type, the distributed stochastic algorithm (DSA) [7], [8], changes are made stochastically by the agents and communicated until a consistent solution is obtained (see II-B.1). In the other type, illustrated by the simple distributed improvement algorithm (SDI) (see II-B.2), the changes are communicated and "voted" on in a distributed fashion before being incorporated into a solution. A third type of distributed algorithm (DCO) involves the communication of state information, or beliefs, between agents to infer their respective role using a constraint optimization method. In those algorithms, agents only broadcast

[1] $\frac{P!}{(P-t)!}$ for distinct teams and $P^t$ for non-distinct teams where agents assume multiple roles.

communications without waiting for responses eliminating deadlocks such as the coordinated attack dilemma[9].

The basic sense-think-act agent loop needs to be modified to augment perception with communication from peers. A distributed sense-think-act-communicate agent loop follows:

1) Read messages from neighbors
2) Sense environment
3) Modify internal state and select next action
4) act
5) send messages to neighbors

In this context, a *round* or *cycle* involving communication consists of executing steps 4 and 5 and then, assuming all messages have been *simultaneously* put in the appropriate channels, steps 1, 2 and 3. In synchronous environments, those two alternate modes, *active* and *passive*, are executed in locksteps. In asynchronous environment, the time required for executing a round depends on the internal clock of the agent. Depending on how fast the internal clock of the agent is relative to the other agents and the environment, a discrepancy between the external world and the internal state of the agent will occur. In those algorithms, exploring is another role that the agent can assume. Since the possible roles are not known apriori but obtained through sensory information, they have to be disseminated as a separate message.

*1) Distributed Stochastic Algorithm:* The DSA algorithm (Algorithm 1) is a stochastic algorithm where each agent changes their assignment *concurrently* if it improves the overall solution quality in terms of constraint violations, according to a probability $p$. The stochasticity of the algorithm should reduce the number of possible messages without affecting the overall performance. The role communicated is the high-level role (e.g. surround the prey $p$ to the north). To avoid cycles, a tabu search is conducted when searching for a new role to minimize conflicts. Agent $i$ selects a new role $r_i$ according to its preference $u_{ij}$ and the number of constraint violations $c$ associated with the new role $j$:

**Algorithm 1** Distributed Stochastic Algorithm(DSA)

```
select p_t, a probability threshold
set initial role to explore
active ←true
while (no termination condition) do
if (active) then
   if (a new role is selected) then
     broadcast the new role to neighbors
   endif
   act according to role
    sense environment
   broadcast prey information to neighbors
   active ←false
endif
sense environment
collect neighbors' new roles, if any
generate a random number r
if (r < p_t) then
  select the next best role
    to reduce conflicts
endif
active ←true
end while
```

$$r_i = argmax_j(\frac{u_{ij}}{c_j}) \quad (1)$$

The best role $j$ according to Eq. 1 that is not tabu is selected. If a new role cannot be obtained, the agent switches to exploration. Each agent maintains a view of the intent of the other agents but does not know or infer their role preferences. The agent stochastically selects a different role if there are conflicts but not necessarily a better role in terms of preference. A role might not have a feasible plan if the location of the prey is unknown and the agent has to switch to exploration until then. A message is sent only if the agent selects a new role or switches to exploration. In a closed and static environment, termination occurs after $N$ consistent rounds where $N$ is the number of neighbors and is assumed fixed. At this stage, the agents have *common knowledge* of their respective roles. Termination entails commitment to a role. In an open and dynamic environment, the agents can only weakly commit to a role and no termination is ensured.

*2) Simple Distributed Improvement Algorithm:* A similar algorithm, based on distributed local improvement[10] and the min-conflict heuristic[11], exchanges messages among neighbors before settling on a role that attempts to find a solution minimizing conflicts (Algorithm 2). In a leader's election approach[9], an external consensus based on preference is sought before settling on a role.

**Algorithm 2** Simple Distributed Improvement (SDI)

```
set initial role to explore
active ←true
while (no termination condition) do
   if (active) then
      if (a new role is selected) then
         broadcast the new role to neighbors
      endif
   act according to role
    sense environment
    broadcast prey information to neighbors
    active ←false
   endif
   sense environment
   selectively collect roles from neighbors
     with maximal improvement
   if (conflicts occurs) then
      select a new role
   endif
   active ←false
end while
```

**Algorithm 3** Distributed Constraint Optimization (DCO)

```
set initial role to explore
active ←true
while (no termination condition) do
   if (active) then
     act according to role
     sense environment
     broadcast local state to neighbors
    active ←false
   endif
   collect neighbors' new information, if any
   estimate possible roles with
     a constraint optimization algorithm
   select role
   active ←true
end while
```

In an open and dynamic environment, a role is tentatively followed without reaching a consensus, keeping the option of readjusting when conflicts occur. Upon conflict, an agent will attempt to change its role only if the role is taken by another agent with a higher preference for the role. As in the DSA algorithm, a tabu list is kept to avoid cycles. If a new role cannot be obtained, the current role is kept. The role communicated is the high-level role (i.e. intent) relative to a uniquely identified prey (e.g. surround prey $p$ to the north) along with the agent's preference for the role. A message is sent only if the agent selects a new role.

*3) Distributed Constraint Optimization Algorithm:* An optimization algorithm can be used in parallel fashion by each agent based on sensed and communicated information from the other agents in the group to autonomously determine which role to assume (see Algorithm 3). It is assumed that the other agents reach the same conclusions because they use the same optimization algorithm[12] and the same payoff function. The Hungarian algorithm[13] is used as the optimization method by each agent. Information necessary to determine the payoff of each role needs to be communicated. Therefore, it is the current local state within the perception range, or augmented with second hand information, that is communicated to the neighbors instead of just the intended role. What is being communicated is a location on the grid.

The Hungarian algorithm, also known as the bipartite weighted matching algorithm, solves constraint optimization problems such as the job assignment problem in polynomial time. The implementation of this algorithm follows Munkres' assignment algorithm[14]. The algorithm is run over a utility matrix of $roles \times agents$.The algorithm consists of transforming the matrix into equivalent matrices until the solution can be read off as independent elements of an auxiliary matrix. While additional rows and columns with maximum value can be added to square the matrix, the optimality is no longer guaranteed if the problem is over-constrained, i.e. there are more roles to be filled than agents.

TABLE I

DISTINCTIVE FEATURES OF DISTRIBUTED ROLE ALLOCATION ALGORITHMS

| | belief* | intent* | preference* | stochastic |
|---|---|---|---|---|
| DSA | | √ | | √ |
| SDI | | √ | √ | |
| DCO | √ | | | |

* Description of what is being communicated

Table I summarizes the main features of the algorithms studied.

## III. EXPERIMENTAL EVALUATION

The experiments were conducted with RePast[15], an agent-based simulation and modeling tool where agents act concurrently in a decentralized manner. Its powerful scheduling mechanism was used to model the asynchronous behavior of the agents.

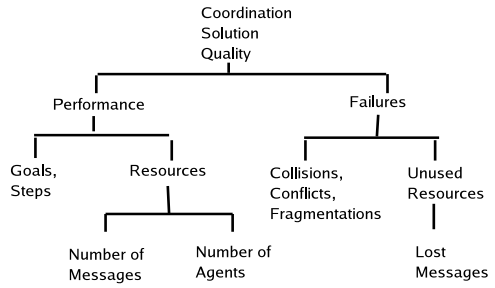


Fig. 3. Taxonomy of Coordination Solution Metrics

### A. Coordination Evaluation

Because coordination is an emergent property of interactive systems, it can only be measured indirectly through the performance of the agents in accomplishing a task where a task is decomposed in a number of goals to achieve. The more complex the task, the higher the number of goals needed to be achieved. While performance is ultimately defined in domain-dependent terms, there are some common characteristics. Performance can be measured either as the number of steps taken to reach the goal, i.e. the time complexity of the task, or as the amount of resources required. An alternative evaluation for coordination is the absence of failures or negative interactions such as collisions or lost messages. Fig. 3 illustrates a taxonomy of coordination solution metrics. To show the scalability of a solution, the evaluation must linearly increase with the complexity of the task[16].

A combined coordination quality measure is defined as the harmonic mean of goals achieved $g$, net resources expanded $r$ and collisions $c$ as follows:

$$g = \frac{\#Preys\,Captured}{\#Preys} \tag{2}$$

$$r = \frac{\#Predators}{log_2(\#Messages\,Received) + \#Predators} \tag{3}$$

$$c = \frac{\#Predators}{log_2(\#Collisions) + \#Predators} \tag{4}$$

$$coordination = \frac{3grc}{gr + rc + cg} \tag{5}$$

### B. Experimental Setup

We experiment with a $n \times n$ toroidal grid and a variable number of predators $P$ and MAFNP preys $p$. We vary the probability of a message to get to its destination according to a normal distribution based on (Euclidean) distance and the communication range $h$ of each agent. The smaller the distance, the more likely a message will get to its destination. To simulate different internal clocks, each predator is given a different schedule interval selected randomly in the [0,1] tick range. The preys are given a fixed schedule set midway at 0.5. The experiments were limited to 100,000 prey cycles. Capture is detected from the prey's point of view when no escape is possible. The probability of changing role was set to 0.7 for the DSA algorithm. To simulate delays in receiving a message due to network traffic congestion, only a fixed number of messages $m$ at a time are kept in the predators' message queue. The messages are processed in temporal order. Unless an updated message arrives, the predators might act on the basis of obsolete information. Messages and sensory information are aged and removed after 3 ticks. A short, reliable prey/predator detection range $r$ simulates the perception range. The path planning routine of the predators is to move horizontally first and then vertically to reach the position indicated by the role allocation. Deadlock among predators can occur because this routine does not take into account other predators resulting in a collision and letting the prey escape. If no role is set, exploration consists of moving one step in a random direction to a location that has not been visited in the past 7 steps.

### C. Evaluation

The experiments illustrate the coordination quality in terms of performance and failures as defined in Eq. 5. The results were averaged over 100 runs.

*1) One prey:* Fig. 4 illustrates the coordination quality obtained on a 10x10 grid with one MAFNP prey for the different predator strategies. The DCO algorithm has an overall significant advantage over the other strategies (t-test p-value was 0.001 compared with the SDI and DSA results). This algorithm also degrades gracefully when communication is

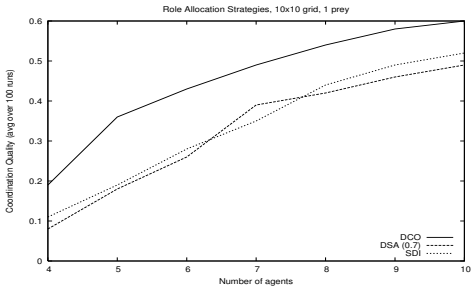


Fig. 4. Comparative coordination quality on a 10x10 grid with one prey

h=4, prange=2,m=1000

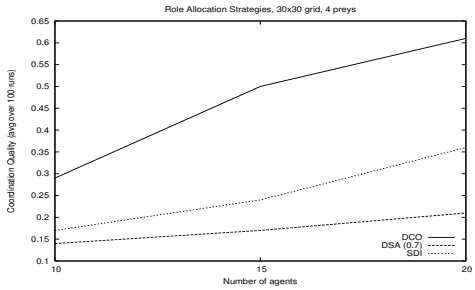


Fig. 5. Comparative coordination quality on a 30x30 grid with 4 preys

h=13, prange=2,m=5000

completely impaired ($h = 0$) since it does not rely on the communication of intent or preference from the other agents to coordinate but assumes identical payoffs. The simplicity of the DSA or SDI algorithms might have some advantage when bandwidth is scarce. In this case, the resources expanded factor in Eq. 3 could be weighted by the load imposed on the network. The experiments also illustrates the diminishing marginal return of adding more team members to the task for the distributed optimization algorithm given a fixed message queue.

*2) Multiple Preys:* Fig. 5 scales up to a 20x20 grid and 2 preys. To decide which team to join (as defined by the target prey) in the DCO algorithm, the agent selects the role in the team that has the maximal sum of preferences rather than maximizing preferences across teams, thereby ensuring team formation. No such team consideration exists for the DSA and SDI algorithms although the SDI algorithm will tend not to disrupt teams close to capture and therefore scales better to higher dimensions.

Fig. 6 shows the change of performance when the task complexity, as measured by the number of goals, is increased given the same resources. The DCO and SDI algorithms experience a degradation of performance while the stochasticity of the DSA algorithm enables it to perform better under increased complexity escaping "needle-in-a-haystack" kind of situations.

*3) Scouts:* Can delegating the exploration task to *scouts* help in the overall task? The scouts' role is to explore and communicate the location of the preys found using similar communication and perception

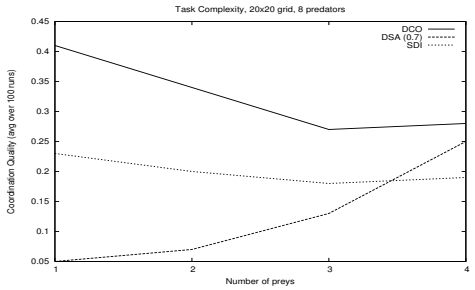


Fig. 6. Comparative coordination quality with increased task complexity

20x20, h=7, prange=2,m=2000

range as the predators. They have no impact on the coordination task except for the implicit exploration role and the additional communication load on the network. Fig. 7 shows the impact of scouts for the different types of algorithms. The impact of scouts is significant only for the SDI algorithm (t-test p-value of 2.18E-17 with the addition of one scout). Under the SDI algorithm, agents keep their role if no conflicts are found but might need to explore if the location of the prey is not known. Scouts help provide this missing information to SDI agents to perform their role.

## IV. Related Work

Research on distributed constraint satisfaction (DCSP) algorithms has concentrated on proving completeness in obtaining a global solution through the communication of neighboring agents rather than responding to dynamic situations[1]. The formalism of DCSP has been extended to dynamic

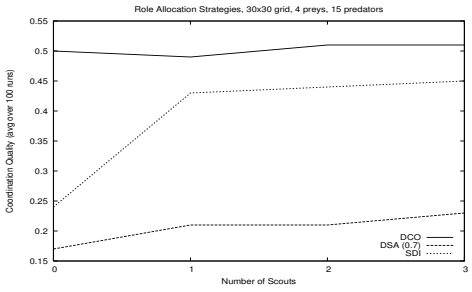


Fig. 7. Scouts' impact on coordination quality

30x30, h=13, prange=2, m=5000

environments and classes of problems have been identified[6] but no results reported for the class of problems such as the prey/predator pursuit game where adaptive approaches without communication have also been applied[5], [17]. Communication-based approaches[18] have however been shown to improve the performance of evolved predators and the experiments confirm and extend those results.

## V. Summary and Conclusions

This work shows that, in open and dynamic environments, inferring intent yields better coordination results than deconflicting communicated intent and the coordination quality obtained far outweighs the communication cost of sharing information. This suggests that learning a common policy for similar payoff and optimization might be more preferable than building a consensus in a dynamic environment. Future work should include extending the distributed constraint optimization method to heterogeneous agents with different initial strategies and preferences. Coupling role allocation with the exploration of role opportunities while maintaining connectivity has also been shown to be critical to the overall performance of multi-agent systems in open and dynamic environments and future work should address this issue as well.

*The authors want to acknowledge useful discussions with Joe Macker and Joe Collins.*

## References

[1] M. Yokoo, *Distributed Constraint Satisfaction*. Springer-Verlag, 1998.

[2] P. J. Modi, *Distributed Constraint Optimization for Multiagent Systems*. PhD thesis, University of Southern California, 2003.

[3] M. Benda, V. Jagannathan, and R. Dodhiawalla, "On optimal cooperation of knowledge sources," Tech. Rep. BCS-G2010-28, Boeing AI Center, Boeing Computer Services, 1985.

[4] R. E. Korf, "A simple solution to pursuit games," in *Working Papers of the 11th International Workshop on Distributed Artificial Intelligence*, pp. 183–194, 1992.

[5] T. Haynes and S. Sen, "Evolving behavioral strategies in predator and prey," in *IJCAI-95 Workshop on Adaptation and Learning in Multiagent Systems*, 1995.

[6] P. J. Modi, H. Jung, M. Tambe, W. Shen, and S. Kulkarni, "Dynamic distributed resource allocation: A distributed constraint satisfaction approach," in *Proceedings of the 7th International Conference on Principles and Practice of Constraint Programming*, 2001.

[7] S. Fitzpatric and L. Meertens, "An experimental assessment of a stochastic, anytime, decentralized, soft colourer for sparse graphs," in *First Symposium on Stochastic Algorithms: Foundations and Applications*, pp. 49–64, 2001.

[8] W. Zhang, G. Wang, and L. Wittenburg, "Distributed stochastic search for constraint satisfaction and optimization: Parallelism, phase transitions and performance," in *Proc. AAAI-02 Workshop on Probabilistic Approaches in Search*, 2002.

[9] N. Lynch, *Distributed Algorithms*. Morgan Kaufmann, 1996.

[10] B. Selman, H. J. Levesque, and D. Mitchell, "A new method for solving hard satisfiability problems," in *Proceedings of the Tenth National Conference on Artificial Intelligence* (P. Rosenbloom and P. Szolovits, eds.), (Menlo Park, California), pp. 440–446, AAAI Press, 1992.

[11] S. Minton, M. Johnston, A. B. Philips, and P. Laird, "Minimizing conflicts: A heuristic repair method for constraint satisfaction and scheduling problems," *Artificial Intelligence*, vol. 58, no. 1-3, pp. 161–205, 1992.

[12] B. P. Gerkey and M. J. Mataric, *RobotCup 2003*, vol. 3020, ch. On Role Allocation in RobotCup. Springer-Verlag Heidelberg, 2004.

[13] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval Research Logistics Quarterly*, vol. 2, no. 83, 1955.

[14] F. Burgeios and J. C. Lassalle, "An extension of the munkres algorithm for the assignment problem to rectangular matrices," *Communication of the ACM*, vol. 14, pp. 802–806, 1971.

[15] N. Collier, "Repast: the recursive porous agent simulation toolkit." Retrieved from http://repast.sourceforge.net, 2001.

[16] E. H. Durfee, "Scaling up agent coordination strategies," *IEEE Computer*, 2001.

[17] S. Sen, M. Sekaran, and J. Hale, "Learning to coordinate without sharing information," in *Proceedings of the Twelfth National Conference on Artificial Intelligence*, (Seattle, WA), pp. 426–431, 1994.

[18] M. Tan, "Multi-agent reinforcement learning: Independent vs. cooperative learning," in *Readings in Agents* (M. N. Huhns and M. P. Singh, eds.), pp. 487–494, San Francisco, CA, USA: Morgan Kaufmann, 1997.