AAAI Technical Report SS-12-01
AI, The Fundamental Social Aggregation Challenge, and the Autonomy of Hybrid Agent Groups

The Effects of Inter-Agent Variation on Developing Stable and Robust Teams

Annie S. Wu
Ramya Pradhan
Gautham Anil
Electrical Engineering & Computer Science
University of Central Florida
Orlando, FL 32816

R. Paul Wiegand
Institute for Simulation & Training
University of Central Florida
3100 Technology Pkwy
Orlando, FL 32816

Abstract

This paper provides a formal analysis of a multi-agent task allocation problem and how variation in agent behavior in the form of response probabilities can be used to build redundancy in the multi-agent system (MAS). In problems where experience is beneficial redundancy provides an MAS with a back-up pool of actors if the primary actors are unavailable. We examine how to ensure a complete team of agents needed for a particular task will be formed, as well as two different ways of determining how to ensure some level of redundancy.

Introduction

One of the expected benefits of multi-agent systems (MAS) over single agent systems is the redundancy that is inherently available in an MAS. In the problem of task allocation, redundancy refers to extra agents beyond the minimum number of required agents that have the capability to perform a given task. Particularly in problems where experience is beneficial, redundancy provides an MAS with a back-up pool of ready actors if the primary actors are unavailable for any reason. This paper provides a formal analysis of a particular single task allocation MAS, examining how variation in agent behavior in the form of probabilistic response tendencies can be used to build redundancy in an MAS.

Response thresholds can be an effective method for allocating tasks among a decentralized team of agents (Bonabeau, Theraulaz, and Deneubourg 1998; Krieger and Billeter 2000; Schelfthout and Holvoet 2002). In this approach, each agent has a threshold at which it will respond to a task stimulus. When the task stimulus falls below an agent's threshold, that agent acts on the task. If all agents have the same threshold value, then all agents will respond at the same time. If, more realistically, agents have different and unique threshold values, then agents will respond gradually, ordered by decreasing threshold values. This implicit ordering of the agents imposed by the agents' threshold values means that the agents with the highest thresholds will be the first to act and have the most opportunities to gain experience. Agents with lower thresholds will have fewer to no opportunities to act and gain experience. Such a system tends to lead to a group of specialists that are very efficient at their tasks and very few, if any, other agents with experience on a given task.

Studies on natural systems such as social insect societies indicate that redundancy in a response threshold system can be achieved "naturally" through probabilistic action (Jones et al. 2004; Ravary et al. 2007; Weidenmüller 2004). If an agent does not act deterministically every time its threshold is met, but rather acts probabilistically, then there is some probability that a high threshold agent will not act, which gives a lower threshold agent the opportunity to act. Over multiple task demands, the system is able to build a pool of back-up agents that have some experience on a task, though they may not have as much experience as the specialists.

We will examine how variation in agent behavior in the form of probabilistic response tendencies can be used to achieve redundancy when an MAS is working on a problem in which experience is beneficial. We assume that the MAS is a response threshold system (Bonabeau, Theraulaz, and Deneubourg 1998) and that previous experience on a task improves an agent's future performance on that task. We examine how probabilistic responses may be used to simultaneously build a redundancy pool and improve team performance (through experience) on a task.

Even relatively simple MASs present some challenges regarding parameterization — they often represent an implementation of a non-linear, complex adaptive system and can thus respond counterintuitively when practitioners adjust parameter values to achieve certain goals. Very often, research into task allocation methods involve a great deal of empirical study (Agassounon and Martinoli 2002; Reijers et al. 2007). One major advantage to our simple, ordered probabilistic framework for task allocation is the ability to characterize the system analytically. In this paper, we provide concrete, theoretically justified advice for how to establish effective parameter values for such a task allocation system for various goals a practitioner has in mind. In particular, we consider the issue of how to ensure a complete team of agents needed for a particular task will be formed, as well as two different ways of determining how to ensure some level of redundancy. Though we focus on a single task allocation scenario here, our goal is to extend these analyses to scenarios that involve multiple tasks.

System Description

Consider a resource mining problem. A team of agents shares a common store of Resource R. Each agent has a threshold below which it will begin mining and collecting additional amounts of Resource R for the store. For example, agent 1 begins mining when the Resource R store is 80% full while agent 2 does not begin mining until the Resource R store drops to 20% full. Each time an agent mines, it gains experience that allows it to mine more efficiently next time. As a result, agents with a high threshold for Resource R are always the first to act when the level of Resource R falls and have multiple opportunities to gain experience and improve their mining ability. Agents with low thresholds for Resource R may or may not have opportunities to act and gain experience. While this does result in the most efficient agents acting on a task, if those agents are busy or lost for any reason, the mining efficiency of the team will drop dramatically due to the low experience level of the remaining agents. If, however, low threshold agents also have occassional opportunities to act, then loss of the high threshold agents for a task will not result in as great a decline in performance. We examine how probabilistic response tendencies may be used to increase action opportunties for low threshold agents.

Given a team of n agents and a task that requires $x : x \leq n$ agents to act, the ordering of the agents defined by their threshold values indicates the order in which agents are offered the opportunity to act on the task. We define a *trial* to be one instance in which the task requires agents to act, a $candidate$ to be an agent that has been offered the opportunity to act, and an $actor$ to be a candidate that chooses to act. Within a trial, agents become candidates in order of their decreasing threshold values. An agent becomes a candidate only if not more than x of the previous candidates have chosen to become actors. Once a candidate, an agent chooses to become an actor with probability s, where $0.0 < s \leq 1.0$ is the *response probability*. A trial ends when either x agents have become actors on the task or when all n agents have been given the opportunity to act on the task.

Let us consider the agents to be ordered from highest threshold (quickest to act) to lowest threshold (slowest to act). If there is no response probability ($s = 1.0$), then the system is deterministic and only the first x agents will act and gain experience. Adding a response probability to the agents' decision making process allows some of the first x agents to choose not to act, thus, providing opportunities for agents beyond the first x to become candidates and possibly actors. Over multiple trials, the system builds a back-up pool of individuals beyond the first x individuals that have some experience acting on the task.

Ensuring Teams and Redundancy

The decentralized, order-based task allocation problem described above involves a number of inherent tradeoffs. For example, as motivated above, it will often be important that over the course of many trials, a variety of agents have some exposure to the task in order to gain experience with the task. One would like to ensure some specified level of redundancy of experienced agents over the course of various trials. In this section, we consider an agent "*experienced*" if it participates in a team on a task in at least one trial.

The obvious way to increase the probability that more agents will gain experience is to reduce the response probability, s. As discussed above, lowering s means candidates earlier in the sequence will not be active, which increases the chance that later agents will have that opportunity. As long as s remains non-zero, the smaller its value, the higher the probability that the final agent will become a candidate.

Unfortunately, there's a catch — lowering s also raises the probability that *too many* agents will reject the task. If this happens, there will be too few agents to make a complete team (there will be fewer than x active agents), and the total number of agents that gain some kind of experience will begin to decrease over repeated trials. Indeed, s should be *as large as possible* in order to increase the probability that we find precisely x agents to collaborate on the task in every trial.

What is needed is a way to establish a value for s that is sufficiently low to make the probability of achieving some level of redundancy as high as possible but sufficiently high to make the probability of making a team in each trial as high as possible. With our task allocation process, it is possible to use traditional bounding techniques for stochastic processes to help determine the response probability that achieves these goals, when they are obtainable.

We divide our discussion into two parts: bounding the probabilities associated with making a team and bounding the probabilities associated with achieving a specified redundancy. In both cases, we will consider the following process, which is equivalent to a single task allocation trial: Assign each of the n agents a "*mark*" with independent probability s, traverse a subset of the agents in order adding each *marked* agent to the team if fewer than x have already been added, then terminate when either there are x agents on the team or all n agents have been considered. Looking at the process this way allows us to focus simply on the number of marks. For our team-forming discussion, let M be a random variable specifying the total number of marked agents, regardless of whether the agents participate in the team.

Lemma 1 *A single trial of the task allocation process will result in $M \leq x - 1$ with $Pr\left\{1 - e^{-\Omega(n)}\right\}$ when $s < \frac{x-1}{en}$. It will result in $M \geq x$ with $Pr\left\{1 - e^{-\Omega(n)}\right\}$ when $s > \frac{x}{n}$.*

Proof: *The expected number of marked agents is $n \cdot s$, $E\{M\} = ns$, since there are n agents to be marked, and each is marked with independent probability s. Let*

$$\delta = \frac{x-1}{ns} - 1$$

and note that

$$(1+\delta)\, ns = \left(1 + \frac{x-1}{ns} - 1\right) ns = x - 1$$

We use Chernoff bounds to bound the probability that there are more than $x - 1$ marks.

$$Pr\{M > x-1\} \quad = \quad Pr\{M > (1+\delta)E\{M\}\}$$

$$
\begin{aligned}
&< \left[\frac{e^{\delta}}{(1+\delta)^{1+\delta}}\right]^{E\{M\}} \\
&= \left[\frac{e^{\frac{x-1}{ns}-1}}{\left(\frac{x-1}{ns}\right)^{\frac{x-1}{ns}}}\right]^{ns} \\
&= \frac{1}{e^{ns}}\left[\left(\frac{ens}{x-1}\right)^{\frac{x-1}{ns}}\right]^{ns}
\end{aligned}
$$

If $x-1 > ens$, this converges to 0 exponentially fast as n grows. Thus $s < \frac{x-1}{en}$ implies $Pr\{M < x\} = 1 - e^{-\Omega(n)}$.

Now consider $\delta = 1 - \frac{x}{ns}$ and note that

$$(1-\delta)\,ns = \left(1 - 1 + \frac{x}{ns}\right) ns = x$$

We use Chernoff bounds to bound the probability that there are fewer than x marks.

$$
\begin{aligned}
Pr\{M < x\} &= Pr\{M < (1-\delta)E\{M\}\} \\
&< e^{\delta^2 E\{M\}/2} \\
&= e^{\left(1-\frac{x}{ns}\right)^2 ns/2} \\
&= e^{-\frac{ns}{2}(ns-x)^2}
\end{aligned}
$$

If $x < ns$, this converges to 0 exponentially fast as n grows. Thus $s > \frac{x}{n}$ implies $Pr\{M \geq x\} = 1 - e^{-\Omega(n)}$. □

Theorem 1 *With overwhelming probability, as n grows a complete team will almost surely be formed when $s > \frac{x}{n}$ and will almost surely not be formed when $s < \frac{x-1}{en}$.*

Proof: *If there are fewer than x marks over all n agents, a complete team of x agents will not be formed, and a complete team can only be formed if there are x or more marks. Noting this, the conclusion follows from Lemma 1.* □

To discuss redundancy, we introduce the concept of a *redundancy factor*, c. This factor is defined such that cx is the total number of agents a user wishes to gain some experience over some number of trials (without loss of generality, we assume that $cx = \lceil cx \rceil$). To ensure this, we need that the ultimate probability that an agent is active, P_i, is not "too small". For this, we examine the probability that the cx^{th} agent is given the opportunity to act. Let M now be the random variable representing the number of marks in the first cx agents. We first describe a bound on s that is *sufficient* to assure a reasonable P_i, then we describe a looser bound that is *required* if we do not want P_i to converge to 0 with team size.

Lemma 2 *In a single trial of the task allocation process, if $s \leq \frac{1}{ec}$, then $P_{cx} = 1 - \Omega(1)$.*

Proof: *The expected number of marks in the first cx agents is $c \cdot x \cdot s$, $E\{M\} = cxs$. Let*

$$\delta = \frac{1}{cs} - 1$$

and note that

$$(1+\delta)\,cxs = \left(1 + \frac{1}{cs} - 1\right) cxs = x$$

We use Chernoff bounds to bound the probability that there are at least x marks in the first cx agents:

$$
\begin{aligned}
Pr\{M \geq x\} &= Pr\{M > (1+\delta)E\{M\}\} \\
&< \left[\frac{e^{\delta}}{(1+\delta)^{1+\delta}}\right]^{E\{M\}} \\
&= e^{-csx} \cdot (ecs)^x
\end{aligned}
$$

So when $s < \frac{1}{ec}$, this approaches 0 exponentially fast with x. Thus the cx^{th} agent almost surely is given the opportunity to act and $P_{cx} = s \cdot \left(1 - e^{-\Omega(1)}\right) = 1 - \Omega(1)$ for constant s. □

Lemma 3 *In a single trial of the task allocation process, if $s > \frac{1}{c}$, then $P_{cx} = e^{-\Omega(1)}$.*

Proof: *Again $E\{M\} = cxs$. Now let*

$$\delta = 1 - \frac{1}{cs}$$

and note that

$$(1-\delta)\,cxs = \left(1 - 1 + \frac{1}{cs}\right) cxs = x$$

We use Chernoff bounds to bound the probability that there are fewer than x marks in the first cx agents:

$$
\begin{aligned}
Pr\{M < x\} &= Pr\{M < (1-\delta)E\{M\}\} \\
&< e^{\delta^2 E\{M\}/2} \\
&= e^{-\frac{x}{2cs}\left(1-\frac{1}{cs}\right)^2}
\end{aligned}
$$

So when $s > \frac{1}{c}$, this approaches 0 exponentially fast with x. Thus the probability that the cx^{th} agent is even given the opportunity to act is exponentially small for constant s. □

Summarizing for a given trial:

- If $s < \frac{x-1}{en}$, a team will almost certainly *fail* to be completed;
- If $s > \frac{x}{n}$, a team *will* almost certainly be completed;
- If $s \leq \frac{1}{ec}$, there is a constant probability that the cx^{th} agent will gain experience
- If $s > \frac{1}{c}$, the cx^{th} agent will almost certainly *fail* to gain experience

Given this analysis, the safest value for s is one that is greater than $\frac{x}{n}$ in order to ensure a team is made, *and* it is less than $\frac{1}{ec}$ to ensure a constant P_{cx}. However, these two bounds often do not overlap, so this is often not possible. Fortunately, we can be a bit coarser in our advice regarding the upper bound. As long as $\frac{1}{ec} < \frac{x}{n}$, there is no good reason to set $s < \frac{1}{ec}$ since doing so would only reduce the expected number of agents that gain experience, and thus the redundancy. Moreover, when P_{cx} is constant with respect to x, the expected number of agents to gain experience will

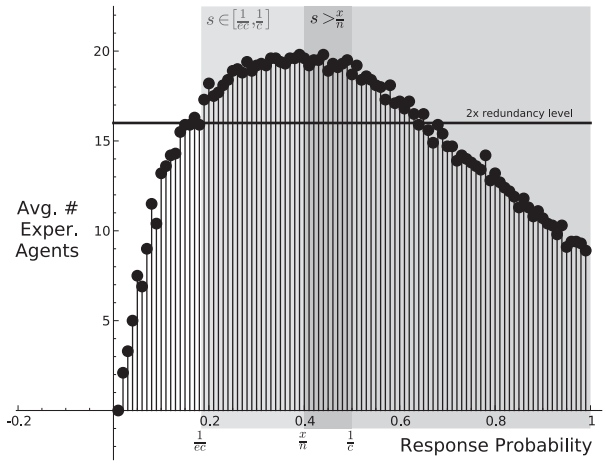

Figure 1: Average number of experienced agents for various values of $s \in [0, 1]$, where $m = 10$, $n = 10$, $x = 8$, and $c = 2$.

typically be larger than cx with a sufficient number of trials because many of the agents *past* the cx^{th} agent will almost have some experience. Consequently, it is more practical to discuss for what values of s our goal *cannot* be achieved: when $\frac{x}{n} > \frac{1}{c}$, we will either have to choose to set s large enough to make a team or small enough to avoid failing to achieve our redundancy factor; we cannot do both. When $\frac{x}{n} \leq \frac{1}{c}$, the most practical advice is to set s to a value slightly larger than $\frac{x}{n}$.

Figure 1 demonstrates how these bounds relate to system redundancy for a system with 10 agents and tasks requiring 6 agents. Each point represents the number of agents with experience in at least one of 10 trials, averaged over 10 simulations. The two shaded regions represent the analytical bounds for ensuring a complete team is formed and a sufficient level of redundancy is obtained ($c = 2$). The darkest shaded region is where these bounds overlap. From this example, it is clear that setting the response probability near x/n is more than sufficient to ensure redundancy.

Optimizing for Redundancy

In the previous section, we showed how simple bounding techniques can be used to give advice about the range of "reasonable" values for the response probability to achieve the goals of forming a complete team while also ensuring some redundancy. This advice has the advantage of being simple and efficient to compute; however, it provides only indirect advice regarding redundancy since it only considers the probability of the cx^{th} agent acting in a given trial, rather than the expected number of agents that act over many trials. Our goal here is to tune the scenario to achieve experience goals after m trials. The first time an agent becomes an actor, that agent is marked as having experience. For our purposes, acting a second time makes no difference. To optimize the experience distribution, we need to first find the probability of gaining experience in m trials for every agent.

The first step is to describe the probability of the ith agent becoming a candidate in a single trial, C_i. This probability is 1 for $i \leq x$ because we need at least x agents. For the rest, they won't become candidates if the team has already been completed, which is governed by the cumulative binomial distribution.

$$C_i = \begin{cases} \sum_{k=0}^{x-1} \binom{i-1}{k} s^k (1-s)^{i-1-k} & i > x \\ 1 & i \leq x \end{cases} \tag{1}$$

As each candidate becomes an actor with probability s, the probability of the ith agent becoming an actor follows.

$$P_i = sC_i \tag{2}$$

Using P_i, we can use the geometric distribution to compute T_i, the probability of being an experienced agent after m trials.

$$T_i = 1 - (1 - P_i)^m \tag{3}$$

This probability distribution T_i over n agents tells about the experience of agents after m trials. This distribution is obviously affected by the parameters x, m, and s — and it is possible to directly optimize s to achieve desired properties within this distribution. Consider one such target distribution Z_i given below. This distribution indicates that we are interested in getting cx trained agents after m trials.

$$Z_i = \begin{cases} 1 & i \leq cx \\ 0 & i > cx \end{cases}$$

We assume x and m are given to us and that we are free to change only s. Even within these restrictions, we envision two different scenarios — one where we need exactly cx agents and another where we need at least cx.

In the first scenario, we minimize the mean squared error to the distribution over all the agents including some additional (say n) virtual agents that are placed after the nth agent. The virtual agents allow us to specify that we want to minimize the chance of not making a team. We can compute the s that solves the first scenario targeting exactly cx agents as follows.

$$s_1 = \text{argmin}_s \sum_{i=0}^{2n} (T_i(s) - Z_i)^2$$

In the second scenario, we minimize mean squared error only over the first cx agents and virtual agents after n. This implies we are interested in getting at least cx agents and additionally want to minimize chances of not making a team. The s that produces the best T_i distribution can be expressed as follows.

$$s_2 = \text{argmin}_s \left(\sum_{i=0}^{cx} (T_i(s) - Z_i)^2 + \sum_{i=n+1}^{2n} (T_i(s) - Z_i)^2 \right)$$

For conditions $n = 100, x = 20, c = 2, m = 10$ and 100 virtual agents, the distributions that result from optimizing for both these strategies are demonstrated in Figure 2. Note the shaded area under the target function Z_i indicating that we want the first cx agents to get an experience for sure. The triangle points indicate the T_i distribution for $s = 0.5912$

optimizing the first scenario while the star points indicate the distribution for $s = 0.4413$ optimizing the second scenario. The star distribution is notably to the right as we are not penalizing additional trained agents. The vertical line at 100 separates the real agents on the left and the virtual agents on the right.

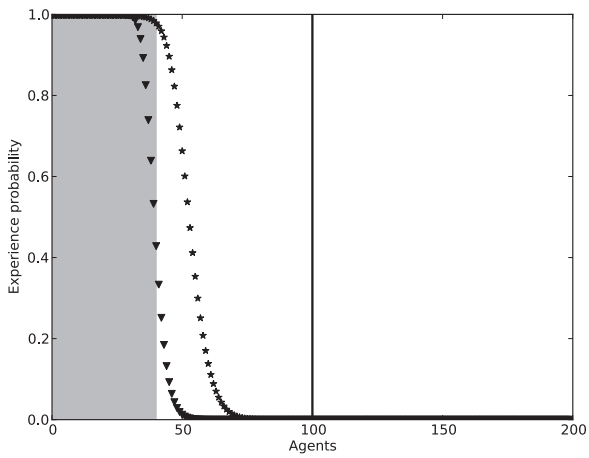

Figure 2: A plot showing the area under the target function as shaded and optimal distributions for two scenarios in triangle and star points. Virtual agents are to the right of the black line.

Empirical comparisons using an agent based simulation yield comparable results. Algorithm 1 shows the pseudocode for our simulation. Using the same parameter settings as above ($x = 20, c = 2, m = 10$ and $n = 200 = 100$ real $+100$ virtual agents), we run the agent based simulation using the s values above for comparison with the data from Figure 2.

Figures 3 and 4 plot the normalized number of times an agent gains experience at least once for $s = 0.4413$ and $s = 0.5912$, respectively. Each point shows the average value and standard deviation over 500 simulation runs, where each simulation consists of $m = 10$ trials. The x-axis shows the number of agents n. The y-axis indicates the normalized value of the number of times an agent gains experience at least once along with the standard deviation. The vertical line indicates the cx^{th} agent. These plots indicate that our empirical results match closely with the theoretical expectations and suggest that our analytical approach provides an effective guide for achieving specific task allocation and experience goals.

Algorithm 1 TISIMULATION$(n, x, s, NumOfSims, m)$

```
for k = 0 to NumOfSims do
    create n agents and intialize their actCount to zero
    for j = 0 to m do
        localCount ← 0
        for i = 0 to n do
            if localCount < x then
                randNum ∈ [0.0, 1.0]
                if randNum ≤ s then
                    increment agent i's actCount
                end if
            end if
        end for
    end for
    {Calculate number of actors (SimActors[][]) in this simulation}
    for i = 0 to n do
        if agent i's actCount ≥ 1 then
            SimActors[k][i] ← 1
        else
            SimActors[k][i] ← 0
        end if
    end for
    {Reset actCount to zero at the end of this simulation}
    for i = 0 to n do
        agent i's actCount ← 0
    end for
end for
{Steps to record number of times an agent acts in NumOfSims}
for i = 0 to n do
    tempSum ← 0
    for k = 0 to NumOfSims do
        tempSum ← tempSum + SimActors[k][i]
        {record if an agent acted at least once in a simulation or not}
        tempDev[k] ← SimActors[k][i]
    end for
    {Normalize tempSum; store normalized value in Ti[]}
    Ti[i] ← tempSum/NumOfSims
    {Obtain standard deviation of tempDev[] and normalize}
    SD ← std_dev(tempDev)
    NstddevActing[i] ← SD/NumOfSims
end for
return Ti, NstddevActing
```

Conclusions

In this paper, we examine the effects of response probability on the number of agents in an MAS that gain experience on a task. We assume that agents are ordered, use response thresholds in their decision making process, and that past experience is beneficial to future decision making. Our goal was to show how fairly simple analytical techniques can be used to help provide very specific guidance for how to determine parameter values in the system to achieve certain experience goals.

We used traditional bounding techniques from stochastic processes to show how to set the response probability to ensure that a complete team is likely to be formed in a given trial *and* a reasonable probability that a specified level of redundancy will be possible. The same method also makes it

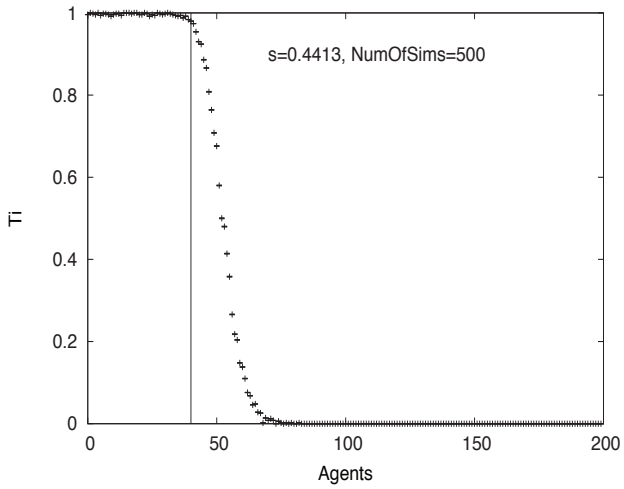

Figure 3: The plot shows T_i the normalized number of times an agent gains experience at least once in 500 agent-based simulations, where one simulation consists of m trials. The x-axis shows the number of agents n. The y-axis indicates the normalized value of the number of times an agent gains experience at least once along with the standard deviation. The vertical line indicates the cx^{th} agent. The curve in the plot is similar to the theoretical curve for $s = 0.4413$.

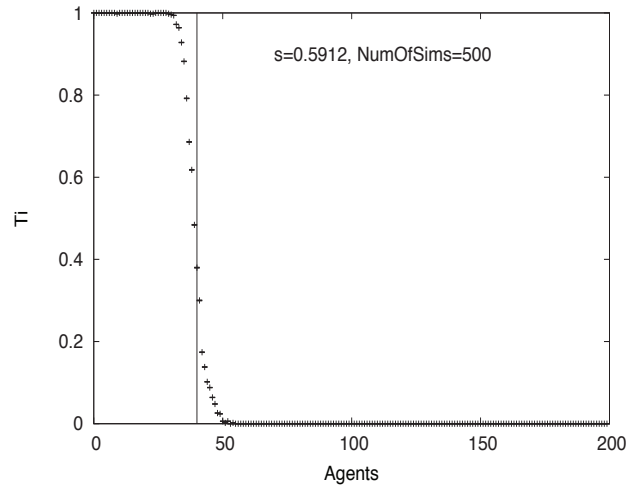

Figure 4: The plot shows T_i the normalized number of times an agent gains experience at least once in 500 agent-based simulations, where one simulation consists of m trials. The x-axis shows the number of agents n. The y-axis indicates the normalized value of the number of times an agent gains experience at least once along with the standard deviation. The vertical line indicates the cx^{th} agent. The curve in the plot is similar to the theoretical curve for $s = 0.5912$.

very clear when it is impossible to achieve both goals. We followed this with a discuss for how to optimize a mathematical model of the task allocation process to achieve specified levels of redundancy over multiple trials.

Future work includes extension to multiple task problems, studies in dynamic environments, and application and testing on a more realistic problem. The work presented here focuses on a single task scenario. More realistically, task allocation problems tend to involve multiple tasks and agents possibly with varying preferences for each task. In addition, real problems often involve task demands and team capabilities that change over time. Extensions to this work will attempt to provide guidance in these more complex environments.

Ackowledgements

This work was supported in part by ONR grant #N0001490911043 and General Dynamics grant #100005MC.

References

Agassounon, W., and Martinoli, A. 2002. Efficiency and robustness of threshold-based distributed allocation algorithms in multi-agent systems. In *Proceedings of the 1st Internationall Joint Conference on Autonomous Agents and Multiagent Systems*, 1090–1097.

Bonabeau, E.; Theraulaz, G.; and Deneubourg, J.-L. 1998. Fixed response thresholds and the regulation of division of labor in insect societies. *Bulletin of Mathematical Biology* 60:753–807.

Jones, J. C.; Myerscough, M. R.; Graham, S.; and Oldroyd, B. P. 2004. Honey bee nest thermoregulation: Diversity promotes stability. *Science* 305(5682):402–404.

Krieger, M. J., and Billeter, J.-B. 2000. The call of duty: Self-organised task allocation in a population of up to twelve mobile robots. *Robotics and Autonomous Systems* 30(1-2):65–84.

Ravary, F.; Lecoutey, E.; Kaminski, G.; Châline, N.; and Jaisson, P. 2007. Individual experience alone can generate lasting division of labor in ants. *Current Biology* 17:1308–1312.

Reijers, H. A.; Jansen-Vullers, M. H.; zur Muehlen, M.; and Appl, W. 2007. Workflow management systems + swarm intelligence = dynamic task assignment for emergency management applications. In *Business Process Management'07*, 125–140.

Schelfthout, K., and Holvoet, T. 2002. 'To do or not to do': The individual's model for emergent task allocation. In *Proc. AISB Symp. Adaptive Agents and Multi-Agent Systems*, 111–115.

Weidenm̈uller, A. 2004. The control of nest climate in bumblebee (*bombus terrestris*) colonies: interindividual variability and self reinforcement in fanning response. *Behavioral Ecology* 15(1):120–128.