Congregation Formation in Multiagent Systems

Christopher H. Brooks and Edmund H. Durfee
Artificial Intelligence Laboratory
University of Michigan
Ann Arbor, MI 48109
({`chbrooks, durfee`}`@umich.edu`*)*

Abstract. We present *congregating* both as a metaphor for describing and modeling multiagent systems (MAS) and as a means for reducing coordination costs in large-scale MAS. When agents must search for other agents to interact with, congregations provide a way for agents to bias this search towards groups of agents that have tended to produce successful interactions in the past. This causes each agent's search problem to scale with the size of a congregation rather than the size of the population as a whole. In this paper, we present a formal model of a congregation and then apply Vidal and Durfee's CLRI framework [24] to the congregating problem. We apply congregating to the affinity group domain, and show that if agents are unable to describe congregations to each other, the problem of forming optimal congregations grows exponentially with the number of agents. The introduction of labelers provides a means of coordinating agent decisions, thereby reducing the problem's complexity. We then show how a structured label space can be exploited to simplify the labeler's decision problem and make the congregating problem linear in the number of labels. We then present experimental evidence demonstrating how congregating can be used to reduce agents' search costs, thereby allowing the system to scale up. We conclude with a comparison to other methods for coordinating multiagent behavior, particularly teams and coalitions.

1. Introduction

In a multiagent system (MAS), self-interested agents must often decide which other agents they want to interact with. The nature of these interactions may vary; perhaps they wish to buy and sell goods, exchange information about their environment, group together in order to exploit scaling effects, or simply benefit from the presence of other agents. These interactions are what makes a society more than just a collection of agents that happen to be in the same location; each agent's reward is dependent upon the agents that it interacts with.

In a dynamic, long-lived system, agents will make this decision as to whom to interact with over and over. Also, as the number of agents in a system increases, the number of potential interactions that a particular agent must consider grows exponentially, since an agent must potentially consider all groups of agents that it might interact with. If multiagent systems are to scale to large numbers of agents, and if these systems are to be dynamic and allow separately designed agents to enter and leave at will, something is needed to save an agent from having to make this expensive computation every single time it needs to interact with other agents. This solution should allow an agent to devote some initial and periodic energy to searching for suitable partners, with the promise of reduced search costs in future iterations. It should also provide a sort of institutional memory, so that like-minded agents can benefit from previous searches performed by other agents.

One way in which human societies have dealt with this problem is through the establishment of *congregations*. Briefly, a congregation consists of a meeting place and the agents that gather there. Human congregations include clubs, churches, marketplaces, university departments, and Usenet newsgroups. In all of these cases, members of these congregations have devoted some upfront cost to organizing and describing themselves so that they (and similarly-minded agents) can reap the long-term benefits of finding preferable agents to interact with without a large search cost, and can also be able to attract new agents with whom they would be likely to want to interact. A congregation allows an agent to localize its search for other agents; rather than considering the entire population as being equally likely to contain agents worth interacting with, an agent can bias its search towards members of its congregation. Since it has had successful interactions with agents from a particular congregation in the past, it can begin by searching only within that congregation. This means that each agent's search problem grows with the size of the congregation, rather than the size of the population, thereby providing a means for open MAS to scale to large numbers of agents. The number of "types" of agents is likely to grow with the number of agents, but at a much smaller rate. In this way, as the population grows, the size of each congregation remains essentially constant, allowing societies of agents to scale to large numbers.

A common problem in MAS is what Davis and Smith refer to as the 'connection problem' [4]. The connection problem is the problem that an agent in an open MAS has when it needs to find another agent to cooperate with, assist it, or interact with. As the number of agents in the system grows, the number of messages that must be sent grows exponentially. One proposed solution to this problem was "focused addressing" [15], in which an agent sends a request to a particular subset of agents that it believes are likely to be able to assist it. Congregating can be viewed as a process by which the knowledge for this focused addressing is instantiated. Once a congregation is in place, an agent can then send coordinating messages within the congregation. As long as congregation size remains relatively constant, the population as a whole can grow without impacting scalability.

Our previous work [1, 2] has provided an introduction to congregations. If congregating is to be a successful technique for scaling MAS, we must understand how easy or difficult it is to form these congregations. We answer this question by defining the concept of a congregation in greater detail and presenting analyses and experiments that describe the problem of how agents self-organize to find the correct congregation. This problem, which we shall refer to as the congregating problem, can be thought of as a distributed search problem. A number of independent problem solvers are simultaneously searching through a space of potential congregations, each with the goal of finding a congregation that allows it to efficiently find the most appropriate agents with which to interact.

The second, and equally important, aspect of the congregating problem is the nonequilibrium behavior of the search. In other words, how difficult is it for an agent to find a suitable congregation? How long does it take, and how much effort or utility must be expended in this search? If an agent must incur a large cost to find or attract a suitable congregation, it may turn out that the effort of congregating outweighs any potential benefits. For example, if the population of agents changes rapidly, an agent may spend all of its efforts maintaining a congregation and never have an opportunity to enjoy the benefit of the reduced search costs. If we are to analyze the benefit of congregations as a means of helping agents find suitable partners to interact with, we must consider nonequilibrium costs and payoffs.

In this paper, we study congregating in the context of the affinity group domain. This is a domain in which agents of different types are trying to collocate in a large space of possible locations. Agents prefer to be with other agents of a similar type, but each agent makes the decision as to where it should locate independently. This domain has parallels to real-life problems in housing and segregation. We have also studied congregating in more competitive domains, in particular information economies, in which producers and consumers are simultaneously trying to locate niche markets where they can buy or sell preferred goods. While that work is not reported on in this paper, we would like to point out that the idea of congregations can be applied to both competitive and cooperative multiagent systems.

We will reserve a comparison of congregating to other existing work on multiagent coordination until section 8, when congregating has been described more precisely. For the moment, hopefully it will suffice to consider congregating as a way for agents in a large population to self-organize into smaller groups without having any knowledge of either the other agents in the population or how they should describe themselves. Once these congregations are in place, agents can then coordinate using whatever mechanism is appropriate.

We begin with a specific definition of a congregation, including characteristics necessary for congregating to be applied to an agent domain. We then introduce a formal model of congregations and show how Vidal and Durfee's CLRI model [24] for analyzing multiagent learning can be used to predict the difficulty of a congregating problem. Section 5 describes the affinity group domain in more detail and formalizes agent preferences in this domain.

Section 6 presents an analysis of the difficulty of the multiagent learning problem, along with experiments showing that the rate of convergence increases exponentially with the number of agents. This problem can be alleviated if congregators can coordinate their decisions; we propose the use of labels as a solution. We show how labelers using a flat label space can make the congregating problem dependent upon the number of labels, rather than the number of agents. This allows the congregating problem to scale with the number of labelers. However, labelers must still potentially search a large space of labels. A hierarchical structure of labels can be exploited so as to allow it to be searched more efficiently, thereby reducing labelers' search costs and allowing the congregating problem to scale to large numbers of agents. In section 7, we return to the issue of congregating as a means of helping agents coordinate their decisions and demonstrate that the introduction of multiple congregations can improve agent payoff. Finally, in section 8 we compare congregating to other methods for multiagent coordination, particularly coalition and team formation.

2. What is a Congregation?

In this section, we introduce a concrete definition of what we mean by a congregation, including features of multi-agent problem domains that must be present for a group of agents to properly be considered as a congregation. We then provide some examples of congregating, discuss the idea of congregating as a distributed search through a space of agent interactions, and illustrate how this can shed some light on the difficulties of congregating in particular domains.

2.1. A Definition of Congregations and Congregating

We begin by providing a definition of congregations and the congregating problem. A congregation is a location and a set of agents that are all gathered in that location. Each agent must interact with others in the course of satisfying its needs. The degree to which these needs are satisfied will depend upon the particular agents it interacts with. The purpose of the congregation is to allow a member to find preferable partners to interact with more easily. The membership of a congregation can be dynamic; agents may join and leave throughout the congregation's lifetime.

Since an agent's satisfaction with a congregation is dependent upon the presence of other agents, and each of those agents' satisfaction is dependent upon the presence of other agents, inducing a particular set of congregations to form can be quite difficult, even if all agents within a congregation would be happy with it once it had formed. In talking about the congregating problem, typically we will be interested in a MAS that is divided into one or more congregations and considering the satisfaction of each member of those congregations. We can then compare MAS with different configurations of congregations, either by using preference aggregation methods or with game theoretic concepts such as Pareto optimality, to assess the relative global desirability of a particular configuration of congregations.

2.2. Characteristics of Congregations

The definition of a congregation provided above is correct, but rather terse and abstract. In this section we provide five characteristics needed for a group of agents to be considered a congregation.

- Individual rationality. Each agent is assumed to have its own utility function. An agent will act solely to maximize its long-term utility, where "long-term" indicates that an agent will take a discounted estimate of future rewards into consideration when deciding with whom to congregate.

  Note that we do not require (nor do we expect to use) any notion of "group rationality." Groups of agents that receive a single lump payment as a result of the group's performance do not fit into our definition of congregations; these are more accurately described by existing work on coalition [19] or team [22] formation.

- Agents may voluntarily join or leave congregations. An essential facet of the congregating problem is that agents are free to join or leave, or refuse to join or leave a congregation at any time they wish. It is this decision problem (which congregation or series of congregations should an agent join?) that is at the heart of this work.

- Another principal idea behind congregations is that an agent's satisfaction with a congregation is dependent upon the other members of the congregation. Since agents congregate in order to satisfy needs which they cannot satisfy alone or to avoid interference in satisfying their needs, it seems reasonable to assume that their satisfaction will depend upon how well these needs are met. If agents are heterogeneous in their abilities to satisfy the needs of a given agent, then that agent will prefer to congregate with those agents which better satisfy its needs over those who do not.

- Agents will have repeated interaction and long-term existence. As noted above, the whole point of developing a congregation is to allow an agent to devote fewer resources to future decisions regarding whom it should interact with. If an agent is making a one-shot decision, there is no value to exploring and learning information to use in future encounters.

- We also assume that agents must expend energy or pay a cost to search for suitable partners to interact with and to advertise their presence to other agents. This cost can vary depending upon (for example) the distance between agents, the number of agents messages are sent to, or the complexity of the message. Advertising messages may also take time to propagate through the population. If an agent is able to costlessly and instantaneously search through the space of all other agents to find suitable partners, then there is no need for it to spend any effort on forming congregations, since the primary function of a congregation is to reduce the aggregated search and advertising cost.

2.3. Examples of Congregations

Congregations are a pervasive feature of human existence. Recently, they have also begun to arise among computational agents within the Internet and the World Wide Web, albeit in a much more *ad hoc* way. In this section we provide some examples of congregations, some with human and some with artificial agents.

One sort of congregation of human agents is a farmer's market. Buyers and sellers of produce come together at a regular time and place. The actual participants may change from week to week, but individuals that want to sell produce know that they will have a high likelihood of finding customers. Likewise, individuals interested in buying produce know that they are likely to find a good deal. Every participant is self-interested, but each requires the presence of some subset of the other participants in order to satisfy its needs. By devoting some initial investment to creating a suitable marketplace and announcing its presence to others (possibly in a distributed, word-of-mouth fashion), participants attract a congregation of like-minded individuals. Even though the population at large may grow, the congregation provides its members with a scalable way to continue to find more preferred individuals to trade with.

Within the Internet, a common type of congregation is a chat group, IRC channel, or Usenet newsgroup, where individuals with particular interests join together to share information about topics of common concern. Within this domain, two features of congregations become obvious: the relationship between a congregation's description and its composition, and the relationship between the congregation's composition and the utility received by participants. For example, a broadly titled newsgroup, such as "comp.ai", might receive a wide variety of participants, posting on a variety of topics. This can be beneficial in that participants are likely to find someone else with similar interests, but also detrimental, since a large group can make it difficult to sort out interesting posts. As the group membership grows, the entrance of new members with tangentially related interests may decrease the newsgroup's utility for current members. Conversely, a group with a too-specific label might not be able to attract enough members to sustain a productive dialogue. Describing a congregation at a level of detail that attracts only the "right" members can be a difficult problem.

A third example is that of constructing the appropriate set of auctions for information services. In the University of Michigan Digital Library Project (UMDL) [6], agents join auctions to contract for the purchase and sale of information, such as scholarly articles or web queries. One decision that has to be made is the determination of the number of auctions to be constructed and the placement of sellers in appropriate auctions. If auctions have too few participants, or sellers whose products are too similar, it might be hard for buyers to satisfy their needs. However, if auctions are too large, then a buyer must evaluate a large number of services that are unrelated to its needs in order to find sellers offering the services it desires. Also, computational cost can become an issue in large auctions, since computing allocations is a computationally difficult problem in combinatorial auctions. The UMDL uses a centralized agent, the Auction Manager Agent [14], to control the creation of auctions. However, in larger systems, a decentralized approach might be more appropriate. This process of selecting the right set of auctions or markets for information goods is precisely the congregating process. Each buyer would like to be in a group with agents that are selling the good it needs at an acceptable price. Similarly, each seller would like to be in a group with buyers willing to pay at least its reserve price for its good. (These needs might be mutually exclusive.) Agents would also like to have some sort of stable structure, so that they do not have to search through the space of all auctions every time they wish to buy or sell. Once again, once an agent is in the "right" auction, the population as a whole can grow without placing an undue computational burden on the agent's decision-making ability, since it can focus its attention on the goods sold in its congregation.

2.4. Congregating as Distributed Search

As was discussed previously, it can be helpful to consider congregating as a form of distributed search. The state space of the search is the set of all possible congregations of agents. As each agent moves from one congregation to another, the search moves from state to state. Of course, no single agent has absolute control over the state transition. This can make a search process difficult to conduct, since particular actions by several agents may be needed in order to move to particular portions of the state space.

Congregating can potentially provide a means of coordinating this transition without central control. If agents tend to return to congregations where they have previously had successful interactions, the configuration of congregations will tend to change more gradually, rather than in wild bursts. The greater an agent's reliance on past history as a predictor of future success, the less likely the configuration of congregations is to exhibit wild swings in congregation membership. However, too great a reliance on past history can prevent agents from conducting a satisfactory amount of exploration. In systems where agents move, a dependence on past history can also lead to errors in evaluating the utility of current actions. This is discussed further in section 4.

Viewing congregating as distributed search also presents us with a useful set of questions with which to evaluate a particular congregating process. For example, we can ask how long it takes the process to reach an "optimal" set of congregations, where optimal may be defined by an external source. We can also ask how efficient a particular state space trajectory to reach a set of congregations was. In other words, do the agents attempting to congregate incur large penalties while congregating as a result of a particular process? If there are several states that a process will converge to, we can compare them either as equilibrium states, asking what their value is to the participating agents, or as the goal states in a search process, asking how beneficial this search trajectory is to each agent in the process. In this paper, we will focus on how long a given search takes to complete and the aggregate profit accrued during the congregating process.

3. A Formal Model of Congregations

In this section we provide a formal model of congregating and the congregating process. This will later be connected to the CLRI model to analyze the difficulty of the congregating problem as characteristics of the problem change.

A congregation c can be described as a tuple $< \alpha, Ag, t >$, where α is the locus, or location where the congregation has formed, Ag is the set of agents in this congregation, and t is the time. A multiagent system can contain many congregations; we will refer to the set C_t of all congregations in a MAS at time t as the *configuration* of congregations.

A congregation has a locational component known as a locus. The set of available loci is denoted $\alpha = \{\alpha_1, \alpha_2, ..., \alpha_i\}$. A locus need not be a physical location; it could also be a newsgroup, a multicast frequency, or a mailing list. The point is that agents congregating at this locus can all communicate with each other.

We consider two sets of agents of different types. The first type of agent is a *congregator*; these are the actual agents who are moving between congregations. The set of congregators is denoted by CA. The second type of agent is a *labeler*. these agents serve the function of market makers or descriptors; they select labels for the purpose of advertising a congregation to perspective congregators. The set of labelers is denoted LA. In some domains, labelers may not be present, so $LA = \emptyset$. We assume that an agent can be a labeler or a congregator, but not both.

Labelers can choose labels from the set $\Lambda = \{\lambda_0, \lambda_1, \lambda_2, ..., \lambda_j\}$. Labels are placed on a congregation as a means of describing it to potential congregators, thereby providing a potential means of coordination. λ_0 has a special meaning as the "null label", which is a label that carries no information. Congregations which are unlabeled are considered to be described with the null label. While semantic disambiguation of terms by different agents is a difficult problem in AI, it is not the focus of our research. Consequently, we assume that all agents share the same semantic meaning for each label.

3.1. Congregators

A congregator wishes to maximize its long-term utility. The underlying assumption here is that a congregator will have many chances to join congregations and therefore wishes to maximize its total satisfaction over all these encounters. As discussed previously, a congregator can benefit in the long term by devoting some initial resources to finding a good congregation. Thereafter, its decision as to whom to interact with is simplified, since suitable partners are occupying the same congregation. For this work, we assume that a congregator can only join one congregation at a time.

We focus explicitly on agents that operate within a utility-theoretic framework. Therefore, each congregator i has a payoff function P_i which maps from a congregation c to a real number: $P_i : C \rightarrow \mathbb{R}$. Since a congregation's payoff to an agent is dependent upon the other agents that are a part of the congregation, a congregator will not be able to fully evaluate a congregation's utility before deciding whether to join it. While selecting a congregation to join, a congregator will rely on an estimate of its payoff, (denoted $\hat{P}_i$) which maps from loci (and the labels placed on them) to the reals: $\hat{P} : \alpha \rightarrow \mathbb{R}$.

Let Λ_j be the set of labels which are offered to congregators (by labelers) in iteration j. Congregations without a label are assumed to be described by λ_0, the null label. A congregator i's decision function δ_i chooses a locus (using Λ_j) which maximizes $\hat{P}_i$.

We assume that there are discrete iterations in which labels are offered and congregators move. However, we do not assume that all congregators or labelers act at the same rate. For example, some labelers may only select a label every 5 iterations, or some congregators may only move every 10 iterations. While this does not provide truly continuous dynamics, it is general enough to capture a wide range of discrete asynchrony in the speed of decision making.

3.2. Labelers

A labeler's problem is potentially more complicated. As with the congregators, a labeler wishes to maximize its long-term utility. It has one decision to make: which label to offer. There are two confounding factors which make this problem difficult: a labeler does not necessarily know the congregators' preferences over congregations (or labels), and so must learn them; and the labeler's payoff for selecting a particular label may also be contingent on the labels offered by other labelers.

Formally, a labeler la has a payoff function P_{la} which is a function of the agents which join its congregation. $P_{la} : \alpha_1 \times \alpha_2 \times ... \times \alpha_n \rightarrow \mathbb{R}$. Since labelers serve as market makers, their payoff function will be correlated with those of the congregators in their congregation. For example, one payoff function would be for an agent to take 10% of the payoff of each congregator in a congregation, providing it with incentive to maximize a congregation's overall payoff. However, a labeler is not able to directly control which congregators join its congregation; it is only able to select labels. Therefore, it is trying to learn a decision function $D_{la} : \Lambda \rightarrow \mathbb{R}$ which predicts the payoff received from offering a particular label. In fact, D also depends upon the labels offered by other labelers, meaning that the labeler is learning a *moving target function* [24]. As other labelers change their offerings, the optimal offering (and therefore the payoff structure) will change.

In this paper, we shall assume that labelers are 0-level learners in the recursive modeling sense [7]. A 0-level agent is one that does not construct a model of other agents and then find a best response to their predicted actions. Instead, the other agents are simply treated as a part of the environment, and accounted for as a sort of noisy signal.

4. Applying CLRI to Congregating

In order to understand what parameters of the congregating problem can make agents' learning problems easier or harder, we need a model that describes different aspects of the problem, how they relate to each other, and how varying an aspect changes the difficulty of the congregating problem. In this work, we have used CLRI as a model for describing congregating.

CLRI is a framework for analyzing multiagent learning that was developed by Vidal and Durfee [24, 23] [1]. CLRI describes how simultaneously learning agents can affect the difficulty of each other's learning problem. Since the congregating problem is one in which each agent is learning which other agents it does and does not want to interact with, this framework is quite suitable. [2]

A full description of CLRI can be found in [25]. We provide a summary here, followed by our extensions to allow it to be applied to the congregating problem.

CLRI is designed to model worlds in which multiple agents are simultaneously learning a decision function, and where each agent's decision function can depend upon the actions of other agents. This leads to a phenomenon known as the *moving target function problem*, where, as one agent changes its behavior due to learning, other agents' decision functions change.

CLRI assumes a world with a finite set of discrete states $w \in W$ and a finite number of actions $a_i \in A_i$ for each agent i, where $|A_i| \geq 2$. Each agent i is learning a decision function $\delta_i^t(w) : W \rightarrow A$ that predicts the right action to take at time t in world state w. The optimal decision function, which predicts the correct action for each world state, is known as the *target function*, and is denoted as $\Delta_i^t(w)$. Agent i's learning problem is to minimize the difference in prediction between δ and Δ. In the congregating problem, A_i^t contains actions which map to the joining of congregations, given the configuration of congregations at time t. For each possible congregation c_j^t, A_i^t contains an action a which indicates that the congregator will join congregation c_j^t. If an agent is a labeler, then A_i^t is the set of possible labels that can be offered at time t. Therefore, the δ described in this section is an instantiation of the δ discussed in section 3.1.

One point to note here is that the CLRI framework is based upon traditional PAC-learning assumptions [13]. In particular, it assumes that, for a given world state, there is one correct action and all the others are equally incorrect. Also, the Markov property must hold; the state that a set of agents are in at time $t + 1$ must depend only upon the state at time t and their decision functions. As with PAC-learning, these assumptions can provide an approximation to the way in which richer, more complex systems behave. We recognize that these assumptions place limitations on the fidelity with which agent systems can be modeled with CLRI; in order to maintain connections to existing work

[1] The name CLRI refers to four of the parameters (change, learning, retention, impact) typically used to characterize a multiagent learning problem.

[2] As a note to readers who are concerned about a conflict between our characterization of congregation as multiagent learning and our earlier characterization of congregating as distributed search, understand that we are of the opinion that learning is a search through a space of possible hypotheses [13].

on computational learning theory, we will maintain these assumptions throughout this paper. Future work will extend CLRI to focus on more complex domains.

In a multiagent world, the difficulty in learning is that an agent's Δ function is dependent upon the actions selected by other agents. As they learn, an agent's Δ can change. The CLRI framework consists of five parameters that are used to quantify this change.

The first two parameters of CLRI are change rate (c) and learning rate (l). Change rate is the probability that an agent will adjust an incorrect mapping of $\delta(w)$ from time t to time $t+1$, and learning rate is the probability that the agent will change the mapping of $\delta^{t+1}(w)$ to be equal to $\Delta^t(w)$. (That is, the agent will know what it should have done at time t in world state w.) Obviously, l must be less than or equal to c. In fact, l is equal to c times the probability that an adjustment will be the correct one for world state w. In the congregation framework, change rate is the probability that an agent will choose a different congregation than it had previously when faced with the same choices, and learning rate is the probability that an agent will be able to say after the fact where it should have gone (and thus make the right choice the next time it is faced with this set of choices).

The third CLRI parameter is retention rate (r), which is the probability that an agent retains a correct mapping ($\delta^t(w) = \Delta^t(w)$). In terms of congregating, this is the probability that an agent that chose the correct congregation will make this same choice again when faced with the same set of alternatives, even if it has changed its mapping of δ for other configurations of congregations.

The fourth CLRI parameter is volatility (v), which is the probability that the function Δ that the agent is trying to learn changes. In other words, the probability that $\Delta^{t+1}(w) \neq \Delta^t(w)$. Note that nothing is said about *how much* Δ changes or what the correct action at time $t+1$ is, only that the correct action does change. In congregating terms, this is the probability that a locus α which was chosen at time t from a given set of loci is no longer the correct choice at time $t+1$ when given the same set of loci to choose from, presumably because other congregators have changed location.

Since volatility can be difficult to determine through observation, CLRI provides a way to derive it using a fifth parameter: impact (I_{ij}). Impact is the effect that agent i's learning has on agent j's target function. In particular, it is the probability that agent j's Δ changes between t and $t+1$ as a result of agent i's δ changing. In terms of congregating, this is the probability that when congregator c_i decides to choose locus α' rather than α, congregator c_j's best choice of congregations changes.

In order to apply CLRI to congregating in any specific way, we need a domain in which these probabilities can be determined. In this paper, we examine congregating using CLRI within the context of the affinity group domain.

5. The Affinity Group Domain

We chose the affinity group domain as an initial domain for studying the congregating problem. An affinity group is a set of agents that all share some characteristic, such as hair color or an interest in agent research. In the simplest case, agents of a particular affinity group want to join congregations that contain other members of their group and avoid congregations containing members of other affinity groups. With more complex types, an agent will prefer to collocate with agents whose type is more similar to its own over agents whose type is less similar. This is a problem that is often studied in sociology and economics, where agents decide where to locate based on some characteristics, such as the race, age, or income of other agents in the different locations. Thomas Schelling was an early pioneer of this sort of model. His research [17] shows how common economic and sociological phenomena, from urban flight and housing prices to insurance coverage, can be modeled with a population of agents of different types who each make self-interested decisions as to where they should locate or which group to join based on the other agents they expect to find there.

The affinity group domain is very simple, but it is one that has several properties that make it useful for studying congregating. The first is the simplicity of agent preferences. Agents evaluate a congregation based only on the characteristic(s) of the other members (as opposed to something about the locus itself), which makes it easy to generate utility functions for each agent. Second, affinity-type problems typically have a set of Pareto-style solutions in which no agent wants to change congregations, given that all of the other agents are remaining still. For example, any state in which every agent from a particular affinity group is in the same congregation and each affinity group is in a separate congregation is equally preferred. This means that we, as evaluators of the system, can identify these states ahead of time and recognize when the MAS is in one of these states. More importantly, it provides the system with a fixed point; once such an optimal state is reached, all agents will remain where they are. This means that we can detect convergence by noticing when no agents wish to switch congregations. It also means that we will not have scenarios where agent A wishes to congregate with agent B, but agent B does not wish to congregate with agent A. In that instance, the two agents would chase each other indefinitely. In other words, preferences are symmetric.

6. The Difficulty of Congregating

In this section, we use CLRI to determine how the different parameters of the congregating problem can make it easier or harder for agents to locate the correct congregation. We begin with the case in which congregators search for the correct congregation without the help of labelers, and then introduce labelers as a coordination mechanism.

6.1. Congregating Without Labels

As a first pass at understanding the congregating problem, we examine the case in which there are no labelers. This means that, for all congregations, the associated label is λ_0, the null label. Congregators therefore can only guess randomly as to which congregation is best for them.

The simplest possible case is when there is only one affinity group and a number of loci. If there is only one affinity group, then all congregators simply want to congregate in the same location. The problem, of course, is that they don't know ahead of time which locus to choose. This is a very simple problem, but it lets us establish some notation and lead into scenarios with multiple affinity groups.

Assume that there are ϕ agents and m loci. Each agent a receives the following payoff:

$$Payoff(a) = \begin{cases} 1 & \text{if } |C(a)| = \phi \\ 0 & \text{otherwise} \end{cases}$$

where $|C(a)|$ indicates the size of the congregation agent a is in. Congregators will search randomly until they find themselves in a congregation with every other agent.

The first question we can ask is how long it will take for a set of ϕ congregators to find each other. The probability that all agents will choose the same locus on any given time step is: $\epsilon = \left(\frac{1}{m}\right)^{\phi-1}$. Therefore, the agents will succeed in finding each other with probability ϵ on the first iteration, $\epsilon(1-\epsilon)$ on the second, and so on. The probability that all congregators have found each other within t iterations is:

$$\epsilon \sum_{i=0}^{t-1} (i+1)(1-\epsilon)^i = 1 - (1-\epsilon)^t. \tag{1}$$

We can then solve for t to ask how long it takes for a set of congregators to find each other with probability p.

$$t \approx m^{\phi-1} |ln(1-p)|. \tag{2}$$

Since the population does not change, once all the agents have converged to the same congregation, they will remain there, since they are all receiving a positive payoff. Also, if all agents have not converged to the same congregation, all of them will be unsatisfied, since they will be receiving a payoff of 0. Therefore, there are m stable states in the global search process (one for each locus). The salient point in this example is that the difficulty of congregating increases exponentially as more loci are introduced, since the number of stable states grows linearly and the number of world states grows exponentially.

We can use the above formulae to determine the corresponding CLRI parameters. We begin with c, the change rate, which is the probability that an agent will change an incorrect mapping. This will happen any time a congregator is not grouped with every other agent of its affinity group, so $c = 1 - \epsilon$. Similarly, l, the learning rate, is the probability that an incorrect mapping will be changed to a correct one. This is simply c times the probability of picking the right locus, which is $(\frac{1}{m})$. r, the retention rate, is 1. When congregators find the right location, they quit moving and so there is no chance of "forgetting." Finally, we have volatility (v), which is the probability that the target function will change. This is the probability that the congregators are not all in the same location multiplied by the probability that the "right" locus changes on the next iteration: $v = (1 - \epsilon)\frac{m-1}{m}$.

The above parameters indicate that as more loci are added, the problem becomes exponentially more complicated. As we saw before, convergence is exponential in the number of agents. Applying CLRI helps us to sort out why this is the case. In particular, v contains the terms $(1 - \epsilon)$ and $\frac{m-1}{m}$. Since ϵ becomes exponentially small as the number of agents increases, volatility approaches $\frac{m-1}{m}$ in the limit. If m is at all large, this will be close to 1, meaning that every agent's actions have a very strong effect on the target functions of other agents. This is what we would intuitively expect, since the problem depends on all agents making the same decision.

Applying CLRI, even to this simple problem, helps us to understand more about why it is that this system takes exponentially long to converge, not just in terms of a series of stochastic events, which is what equations 1 and 2 are, but in terms of agents that are trying to learn the correct action. In agent terms, the learning problem is difficult because volatility is very high for each pair of agents, meaning that each agent's decision function is highly dependent upon every other agent. This is a different sort of analysis than one which treats agents as passive objects, and will help us to find ways that agents can learn more quickly in section 6.2. Note also that these analyses are worst-case; the solution space is very sparse, and congregations containing only part of an affinity group yield no payoff. While this is nice for performing analyses, it is perhaps overly pessimistic. In section 7, we will expand upon this and examine how congregating can be used to improve overall performance.

6.1.1. *Multiple Affinity Groups*

Of course, a system with one affinity group is of limited interest. Consider the case where there are g affinity groups of congregators and each group is of size s, for a total of gs congregators. Again, there are a total of m loci. This means that there are m^{gs} different world states. Once again, each congregator wants to join a congregation that maximizes its payoff. Let us modify the payoff function slightly.

$$Payoff(a) = \Sigma_{c \in C(a)} \frac{Similarity(c, a)}{|C(a)|} - \tau \tag{3}$$

where $C(a)$ is the congregation an agent is in (and $|C(a)|$ is the number of agents in this congregation). $Similiarity(c, a) : A \times A \rightarrow [0, 1]$ is a function which takes two congregators as input and returns a number between 0 and 1 indicating the congregators' fractional similarity. In addition, each agent has a threshold τ which is subtracted from this sum. This threshold can be interpreted either as a computational or production cost, if agents must pay for their resources, or as a form of satisficing. By tuning τ, we are able to control the fraction of congregations an agent is satisfied with. If τ is near 0, almost any congregation yields positive payoff, whereas if τ is 1, only a congregation consisting entirely of an agent's own affinity group will yield nonnegative payoff. To remain consistent with the PAC-learning assumption that there be only one correct action in a given world state, we will let *Similarity* equal 1 if agents are in the same affinity group and 0 otherwise. In addition, to make non-optimal actions undesirable, we set τ equal to $\frac{s}{s+1}$. This assures that agents will only receive a positive payoff and therefore stay still only in the case where they are in a congregation consisting solely of their own affinity group. These assumptions will be weakened in section 7.

We are now able to determine the CLRI parameters. This problem is quite complicated, since there is such a large number of combinations of congregators. The first parameter is c, the change rate. A congregator will change an incorrect mapping any time it is not in the correct congregation, which, if all agents move randomly, will happen with probability $c = 1 - ((\frac{m-1}{m})^{(g-1)s})$. This is the probability that a congregator is not in a congregation consisting solely of members of its affinity group.

The second parameter is the learning rate (l). This is simply the change rate c times the probability of randomly moving to the best available congregation, which is $(\frac{1}{m-1})$, or $(\frac{1}{m})$ if an agent may return to the congregation it was just in.

The third parameter is the retention rate (r). A congregator will retain a correct mapping if it is in the correct congregation and no congregators of another type arrive. In other words, $r = (\frac{m-1}{m})^{(g-1)s}$.

The fourth parameter is impact (i), which is the effect that agent b's decision has on agent a's target function. Agent b can affect agent a in two ways: if it is different and moves into agent a's congregation, or if it is the same and it moves out. The first case will happen with probability $\frac{1}{m}$. In the second case, a like agent will move out if there is a different agent in the congregation, which happens with probability $1 - \frac{m-1}{m}^{s(g-1)}$.

Determining closed-form solutions for the time to convergence and the probability of convergence is more difficult in this scenario, since it is possible for a subset of congregators to settle early on and wait for others to find them. In other words, the probability of convergence is dependent on the entire past history. However, if we examine the change rate and impact of agents from different affinity groups, we can see that the problem's complexity lies in the fact that both terms contain exponents dependent upon the number of agents: $(s(g-1))$. Once again, the difficulty of each agent's learning problem comes from the fact that its decision function can be affected by so many other agents. As the number of agents increases, the volatility of each agent's decision function increases. Similarly, we can see that each agent's change rate is very high as the population grows. This shows us that the difficulty of the problem stems from the fact that each agent is rapidly changing its decision function, and each of these changes affects the decision functions of other agents in the system. This suggests ways to reduce this complexity: reduce the agents' change rate, or reduce the impact that each agent's decision has on the other agents around it.

Since it is difficult to analytically determine the time needed for a system with multiple affinity groups to converge, we performed some simple experiments to develop an estimate. Each agent had a τ of 1. Using 3 loci, and 3 affinity groups of three congregators each, we begin by correctly fixing 8 congregators and having 1 move. We then fix 7 and allow 2 to move, and so on. Agents move randomly between congregations until they settle on a locus where there are only members of their own affinity group (i.e. their payoff was 1). They then stay there unless "pushed out" by the arrival of a congregator from a different affinity group. Results (averaged over 50 trials) are shown in Figure 1.

As predicted in the analysis, the time needed for the agents to stabilize increases exponentially as the number of congregators allowed to move increases. Since the solution space is so sparse, it quickly becomes very difficult for congregators to find each other. As we discussed previously, the two ways to reduce the problem complexity are to reduce the change rate or to reduce the impact that agents have on each other. One way in which agents can reduce their impact on each other is to coordinate their decisions; for example, all agents of one affinity group in a congregation can decide to move to the same location. This suggestion implies that they can communicate and negotiate; in the

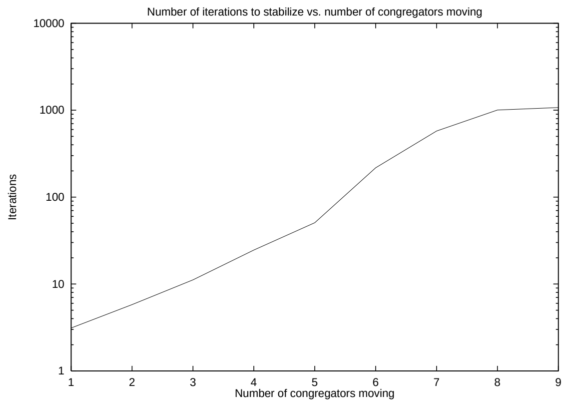

Figure 1. Iterations to stabilize as a function of the number of congregators moving. Note that the y axis is log scale.

following section we will show how this coordination can be achieved without communication between congregators.

6.2. Introducing a Flat Label Space

As we noted earlier, if agents are able to label the loci they are at and attract like members in that way, the congregating problem can potentially become much easier.

In this situation, the congregator's decision function becomes trivial; simply move to the congregation with the label that most closely corresponds to its affinity group. If multiple labels match equally well, choose one at random. If no such label exists, move randomly. In this case, it is the labeler's problem that is the interesting one. Each labeler must choose a label which will attract an affinity group. The value of its decision will be influenced by the choices of the other labelers. Therefore, each labeler wishes to attract a distinct affinity group. We begin with a flat label space. That is, each label potentially refers to at most one affinity group. In section 6.3, we extend these results to consider the case of hierarchical labels.

Let us assume that each labeler is neutral as to which affinity group it wants to attract. It derives utility from having a successful congregation; that is, one in which all congregators in its group are of the same type. (Perhaps it collects a portion of their payoffs.) As in the previous section, we will assume that labelers have no memory regarding the payoffs from labels other than the current one. A labeler simply makes the decision as to whether to keep or discard the current label. We retain the notation used in the previous section, so g indicates the number of groups or types. Additionally, λ indicates the number of labels and m the number of labelers. (We will assume that there are at least as many labels as groups. If this is not the case, it may be more useful to treat the system as one without labels.)

Within this framework, c (the change rate) is 1. If a labeler has an incorrect label, it will always change it. l, the learning rate, is the probability that the labeler will select an unused label, which is $\frac{g}{\lambda}(\frac{\lambda-1}{\lambda})^{(m-1)}$. As the number of labels gets very large, the first half of this term goes to zero and the second half to one. In this case, nearly all the labels do not correspond an affinity group. If we differentiate l, we find (after much reduction) that $\frac{dl}{d\lambda}$ is maximized when $m = \lambda$, and there is one label for each labeler.

r is also 1, by definition; if a labeler has a correct label, it will keep it. Volatility (v) is the probability that the target function will change, which is the reciprocal of the learning rate, since the only way the target function will not change is if all other agents hold still. Therefore, maximizing the learning rate will minimize volatility.

Once again, determining the time needed to converge is non-trivial, since one labeler's decision may aid another. In fact, what typically happens is that one labeler will attract an affinity group, who will then hold still and make the problem easier for other labelers. Therefore, to determine the time needed to converge with probability p, one must determine this probability for each possible history and then aggregate these probabilities appropriately.

By applying CLRI to the problem, we are able to see that labeling will be most effective when the number of labels is close to the number of labelers. Also, as was pointed out in the beginning, this assumes that there is at least one distinct label for each group. This all implies that the difficulty of the problem increases with the number of labels (or affinity groups), rather than the number of congregators, which is a substantial improvement.

Once again, we used experiments to develop an estimate of convergence time. In this experiment, we began introducing labelers for each locus. A labeler could choose which affinity group it wanted to attract; its decision was independent of the other labelers. Congregators would automatically go to a group with the corresponding label (or choose randomly between offerings if more than one labeler offered the same label) or move randomly if the label for their group was not offered. Each congregator received a payoff according to equation 3. The labeler's decision was analogous to the congregators' in the previous experiment: if the label chosen was successful (that is, it attracted a complete affinity group with no extra agents) it would continue to use that label. Otherwise, it would select a new label at random.

We began with a baseline experiment with no labelers. This is the same as allowing all congregators to move in the first experiment. We then added one labeler, then two, and then three. Results are shown in Figure 2.

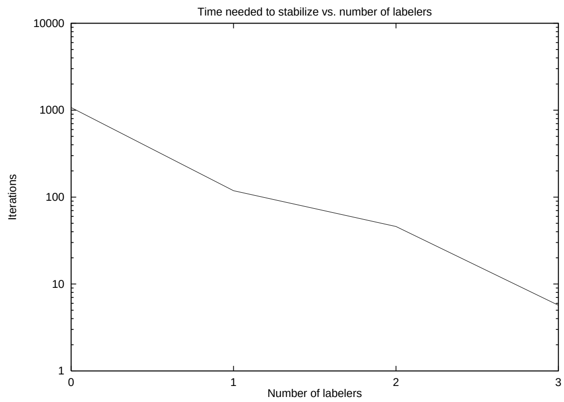

Figure 2. Iterations to stabilize as a function of the number of labelers. Y axis is log scale.

There are a few things to notice here. The first is that the problem becomes trivial with three labelers. They settle on a solution almost immediately. Even with only two labelers, the problem becomes very easy; the congregators which don't have a label to attract them are able to settle on a common locus relatively quickly.

The slightly more subtle point is that adding one labeler produces the same effect as holding two congregators steady, and adding two labelers produces the same effect as holding five congregators steady. The label produces a shared decision amongst all members of the corresponding affinity group; they will all move to the same congregation, and so in terms of convergence and impact they can be considered as one agent. This is a big savings; as was noted previously, a major difficulty with the congregating problem is the impact that one agent's decision has on another. If all agents in an affinity group make the same decision, then they will produce the same amount of impact on an agent of another affinity group as a single agent in a system without labelers. As more affinity groups are introduced, the potential benefits from introducing labelers will be even greater, since a system with labelers will increase impact linearly (with each group added) rather than exponentially (with each group added *times* the number of agents in each group) in systems without labelers. As we saw in the CLRI analysis, volatility is now a function of the number of groups or labels, rather than the number of agents. By coordinating their decisions, members of a particular affinity group are able to simplify the problem both for themselves and for other agents (which reduces other agents' impact on their decision functions). In this case, labeling is used as a mechanism to provide this coordination.

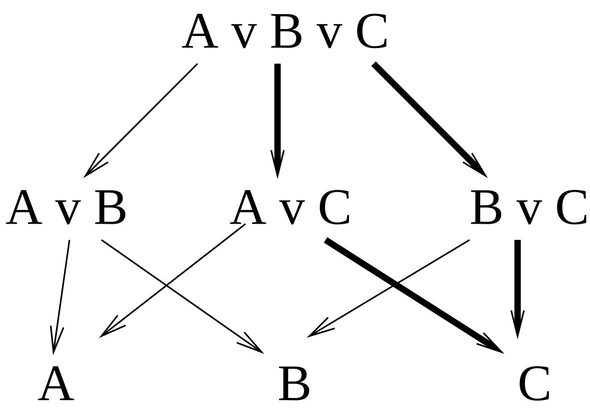

Figure 3. All possible abstract and base labels of a three-label system. Heavy arrows indicate the sub-lattice of labels which cover the base label C.

As was discussed previously, the benefit of labeling is maximized when the set of labels is the same size as the number of affinity groups. If the set of labels is much larger, then using labels to attract a congregation becomes more difficult, since labelers will often choose labels that do not correspond to any affinity group. One way around this is the introduction of hierarchical labels, where some are more general than others.

6.3. Adding a Hierarchy of Labels

One common form of hierarchical labeling, which makes sense within the affinity group domain, is the introduction of abstract labels that can potentially attract more than one affinity group. This is in contrast to a base label, which corresponds to at most one affinity group. When there are many more labels than affinity groups, using abstract labels can make sense; they allow labelers to have a better chance at attracting congregators. In this paper, we restrict ourselves to abstract labels that are the logical OR of simpler labels. The corresponding abstract and base labels form a lattice, as seen in figure 3.

A labeler that can select an abstract label has a tradeoff to make: a more abstract label is more likely to attract congregators, but if too many affinity groups are attracted, their members will not be happy with the congregation and will leave. Also, we assume that a congregator will choose a more specific label over a more general one, since the more specific label will be more likely to contain a higher percentage of congregators in its affinity group, making too-general labels unuseful. The larger the number of labels is relative to the number of affinity groups, the higher in the hierarchy a labeler should select labels. We will now outline this more formally in CLRI terms. We begin with some terminology that will aid in applying CLRI. We refer to the *level* of a label in a hierarchy as the label's height in the hierarchy. In figure 3, the label A is at level 1 and the label $A \vee B$ is at level 2. More general labels have a higher level.

As was mentioned above, one danger of selecting an abstract label is the selection of a similar, but more specific label by another labeler. The labeler that offers the more specific label is *covering* the other labeler. For example, assume there is one affinity group, labeled A. If labeler 1 chooses the label $A \vee B \vee C$ and labeler 2 chooses the label $A \vee B$, labeler 2 covers labeler 1, since all the congregators in group A will choose labeler 2, as its label meets their requirements more specifically.

We will also refer to a label *matching* an affinity group. A label matches an affinity group if congregators in that group would pick this label, given that no other label covers it. For example, if we had labels $\{A, B, C, D\}$, $A \vee C \vee D$ and $A \vee B$ both match group A.

The primary calculation that needs to be made in order to calculate the various CLRI parameters is the probability of a labeler choosing a successful label. A label is successful if it matches at least one affinity group, and it is not covered by any other label. We begin by determining the total number of possible labels. Let g be the number of affinity groups, m be the number of labelers, and l be the number of base labels, where $g < l$. The total number of labels is: $\sum_{i=1}^{l} \binom{l}{i} = 2^l - 1$. If a labeler chooses a label at level l', it will have a probability of $g(1 - \frac{(l-l')}{l}) = g(\frac{l'}{l})$ of matching at least one affinity group. As the label becomes more abstract, the chances of matching an affinity group increase. However, the chances of being covered by another labeler also increase. For example, in figure 3, if labeler 1 chooses label $A \vee B \vee C$ and the affinity group is truly of type C, any labeler who chooses a label in the sub-lattice from C to $A \vee B \vee C$ (indicated with the dark lines) would cover labeler 1.

Given that a labeler has chosen a successful label at level l', the probability that another labeler will cover it is:

$$\begin{aligned} & m(1 - \frac{\sum_{i=1}^{l} \binom{l}{i} - \sum_{i=1}^{l'-1} \binom{l'-1}{i}}{\sum_{i=1}^{l} \binom{l}{i}} = \\ & m(1 - \frac{2^l - 1 - 2^{l'-1} - 1}{2^l - 1}) = \\ & m(1 - (1 - \frac{2^{l'-1} - 1}{2^l - 1})) \approx \\ & m(2^{l'-l+1}) \end{aligned} \tag{4}$$

As a labeler chooses a label with a higher level, its chances of being covered increase exponentially. We can now determine the probability of success as the probability of matching and not being covered. This is:

$$Pr(success \wedge \neg covered) = m^2(2^{l'-l+1})(\frac{l'}{l}).$$

Note that the probability of matching grows with the ratio of l' and l, whereas the probability of being covered grows with their difference.

Once we know this probability, the CLRI parameters are straightforward to determine. As above, change rate and retention rate are 1. Learning rate is simply the probability of success, which was determined above. Impact is the probability that a particular labeler will cover a successful label. This is $2^{l'-l+1}$ (from equation 6). Volatility is the probability that any other labeler will cover a successful label, or m (the number of groups) times impact, or $m2^{l'-l+1}$.

Using CLRI to analyze this problem helps us to identify both what is easy and what is difficult about this problem. We see that learning rate is influenced by both the number of other labelers and the size of the hierarchy. For a single labeler, it is easier to cover more groups by choosing a higher-level abstract label, but when multiple labelers are learning and choosing labels at different levels, it becomes easy for one labeler to cover another. Once labelers have all settled on the same level of abstraction, volatility decreases and learning rate increases.

Once again, convergence time was determined experimentally. The addition of hierarchical labels also adds an additional decision for the labeler: whether to offer more or less abstract labels. Labelers receive a positive fraction of the total payoff of the congregators in their congregation. We chose a very simple strategy for the labelers. A labeler that receives no payoff will attempt to abstract, assuming that it has not matched with anyone. A labeler that receives a less than maximal payoff will choose a less abstract label, assuming that its generality has attracted too many affinity groups. Finally, a labeler that receives the best possible payoff will stay put.

In this experiment, we again considered a scenario in which there were three labelers, three affinity groups of three congregators each, and three base labels. We then held the number of labelers and congregators fixed and introduced extra base labels, up to a maximum of 20. The median number of iterations needed for the labelers to each attract an affinity group (over 100 trials) is shown in figure 4.

We used the median here rather than the mean since the experiments had a large standard deviation. Notice that the number of iterations needed for the system to stabilize increases linearly as more labels are added. Examining the individual runs indicates that, in a typical run, each labeler tended to use about $\frac{1}{3}$ of the available labels. Once each labeler added enough base labels to have this many in their abstract label (each started with one), they were then reduced to a problem very similar to the flat label space presented in section 6.2 where g labelers were choosing from among g labels.

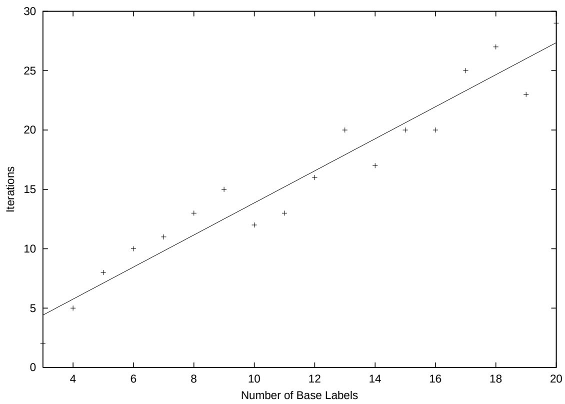

Figure 4. Iterations to stabilize as a function of the number of base labels.

In general, it appears that allowing labelers to offer hierarchical labels and, in particular, to exploit the structure of such a hierarchy when searching for appropriate labels, makes finding optimal sets of congregations much easier.

7. Using Congregations to Improve Net Payoff

The previous section's analytical results are useful for two reasons: they allow us to understand what is difficult about the congregating problem by decomposing it into the decision functions of individual agents, and they suggest ways in which the difficulty of the problem can be ameliorated. However, they also have the effect of obscuring the usefulness of congregating as a way of reducing search cost and thereby improving net payoff to the congregators. This is an artifact of our assumption that τ is 1, meaning that agents are only happy with a small set of congregations. In this section, we relax this assumption and show how congregations can be used to improve agents' overall satisfaction.

Recall the payoff function described in equation 3. It says that an agent receives a fractional payoff equal to the average similarity between it and the other agents in its congregation, less τ. An alternative but equivalent explanation for this function is that, on each iteration, each agent selects another agent from its congregation at random to play a coordination game with. The payoff for this game to each agent is the similarity between the agents, measured on $[0, 1]$. In expectation, this is the same payoff function, yet it provides a clearer illustration of how congregating can be useful: by increasing the expected payoff from this game.

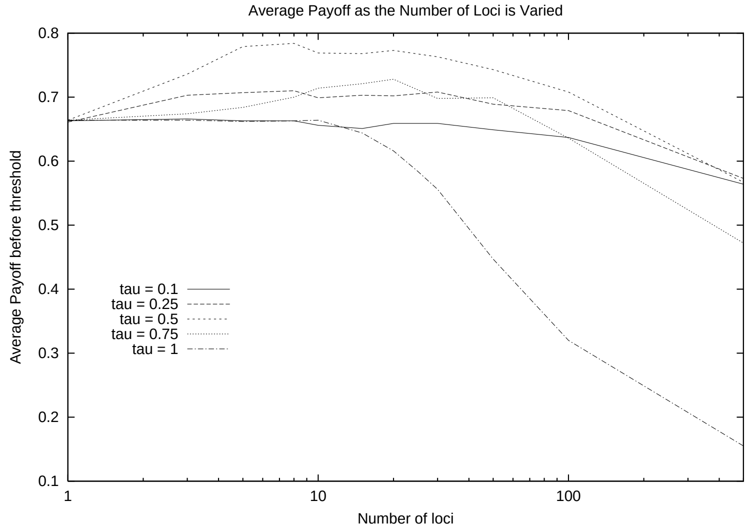

Figure 5. Average profit 50 congregators from 10 affinity groups, averaged over 100 iterations for different numbers of loci as τ is varied between 0 and 1. X axis is log scale.

In order to quantify the relationship between congregating and system performance, we performed experiments with 50 congregators divided equally into 10 affinity groups $\{g_1, ..., g_{10}\}$. The similarity between group i and j is $(1 - \frac{|j-i|}{10})$. No labelers were used, and the number of loci were varied. Unlike in the previous section, τ was also varied between 0 and 1. This provides an illustration of the way in which the solution space attenuates as agent thresholds increase. (All agents have the same τ.) As in section 6.1, congregators moved if their net payoff was negative. Results are averaged over 10 runs of 100 iterations each.

Figure 5 shows the results of these experiments. The x axis indicates the number of loci, and the y axis is the pre-threshold payoff per congregator, averaged over the entire run. (That is, τ has not been subtracted from this total.) Several results stand out from this experiment. First, we see that the benefit of congregating differs as the threshold is varied. When $\tau = 0.1$, agents are happy with almost any interaction, and so there is no need for them to separate into different congregations. As τ increases to 0.25, 0.5 and 0.75, we see a clear benefit to congregating. For each curve, there is a number of loci (near 10) in which average payoff is maximized. When $\tau = 1$, congregating is not able to help much; at this point, the solution space is so sparse, and the threshold so high, that the chances of a congregation forming that is stable enough to hold together and attract more members is very low. Congregating

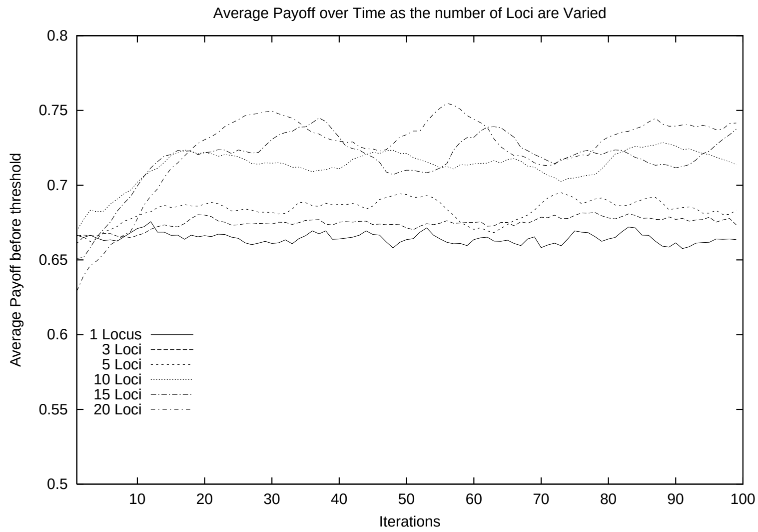

Figure 6. A moving average of average profit over time as the number of loci are varied. $\tau = 0.75$. Each point is an average of the profit for the 10 closest time points, which smoothes the curve. Experiments involved 50 congregators from 10 affinity groups, averaged over 100 iterations.

is most effective when agents of a particular affinity group are able to form a "critical mass" at a locus and then attract others from their group. Second, we can see that, for all thresholds, there is a point at which the number of loci is too large, thereby decreasing average payoff. With a large number of loci, it again becomes difficult to form that critical mass needed for a congregation to form and persist. Another point, which may be more difficult to glean from this summary, is that multiple loci allow the agents to settle into a stable configuration. To see this, consider $\tau = 0.75$. Recall that the y-axis in figure 5 indicates payoff before the threshold is subtracted. Since this is averaged over 100 iterations (including initial learning) when there are 20 loci, this indicates that the system is able to reach states in which agents are receiving positive profit. Figure 6 illustrates this more clearly, showing a moving average for $\tau = 0.75$ as the number of loci is varied. This figure shows that congregating allows agents to improve their payoff, even beyond a relatively high threshold. Of course, there are some agents in the system below the threshold, and so they will change congregations, thereby changing the average profit and producing the cyclic behavior we observe.

This experiment demonstrates that, if the system designer knows something about the congregating population (in this case the number of affinity groups) he can engineer a system with an appropriate number of loci and allow the congregators to self-organize into a configuration that stabilizes to yield higher average payoff than would be achieved if the agents were all selecting other agents within the population as a whole. It also demonstrates that congregating is most effective when agents have some preference over whom they interact with (as opposed to when $\tau = 0.1$), but the dynamics of congregation formation allow for a critical mass of agents to collect together and attract similar agents, as in $\tau = 0.25, \tau = 0.5$, and $\tau = 0.75$.

8. Related Work

In this section, we compare congregating to other methods for coordinating self-interested agents, particularly coalition formation and team formation. It is worth noting that these methods can be combined; congregations are a way for large groups of agents to self-organize into smaller groups. Once these congregations have formed, coalition or team formation methods can be used to allocate tasks or determine relationships *within* the congregation. In this way, these algorithms can scale to larger numbers of agents.

One of the first methods for coordinating multiple agents was Davis and Smith's Contract Net [4], which provided a protocol by which an agent could locate other agents who would perform tasks for it (the connection problem). As discussed earlier, one problem with Contract Net is the number of messages sent as the number of agents increases. Focused addressing, as implemented using congregations, is a way of solving this problem.

Matchmaker services (sometimes referred to as directory services or yellow pages) have also been suggested as a way for agents to solve the connection problem [5]. In these systems, a set of directory agents handle the connection problem by having agents register their capabilities. When agents need to have a task performed, the matchmaker then suggests an appropriate agent. Matchmaking systems, as well as systems such as RETSINA [21] which use middle agents to coordinate the connection problem, assume that each agent can describe its capabilities and knows the sorts of agents it wants to interact with. In a more general system, the best way to describe oneself might depend upon the other agents that are in the system. Congregating provides a way of sorting agents into groups so that they can more easily determine how to describe themselves.

Congregating is also related to multiagent learning problems; multiple agents are learning at the same time, and the global state of the system is an important factor in determining the utility of each individual agent. The notion of coordination as multiagent learning has also been discussed by Sen, et al [18]. In their paper, cooperative agents learned to coordinate actions to achieve a joint goal. Hu and Wellman [9] present a game-theoretic model in which self-interested agents are able to coordinate and each learn a Nash equilibrium strategy. A primary difference between those works and this paper is that the congregating problem focuses on a decision as to *whom* to interact with, rather than what action to choose.

The construction of teams of agents that collaborate to pursue some shared or common goal is a well-studied field of research within MAS. Singh, Rao, and Georgeff [20] define a team as a group of agents who are constrained to have a common goal. One notable example of research on teams and team-oriented programming is that of Tambe [22]. His STEAM framework implements a joint-intentions model of problem solving [3], and demonstrates how groups of agents can be dynamically constructed to work together at problem solving. Similar approaches to multiagent coordination can be found in the work of Jennings [10], who also uses joint intentions, and Grosz and Kraus [8], who use an abstraction known as SharedPlans.

All of these approaches differ from congregating in that they explicitly require agents to share some sort of joint goal to be achieved. As a result, agents' reward or utility is dependent upon the performance of the group as a whole. In contrast, congregating makes no such assumption. All agents are presumed to be completely self-interested. Each agent's utility function is completely local, and an agent will cooperate with other agents only as long as such cooperation increases its utility. Note that this does not prevent the formation of teams so long as team formation benefits all members of the team; the point is that agents cannot be assumed *a priori* to be cooperative and only interested in team goals.

A more self-interested approach to multiagent interaction and coordination is the coalition. Coalitions have been discussed both within the context of economics [12] and multi-agent systems [19, 11, 16, 27]. While some details vary among researchers, the standard definition of a coalition is a group of agents that have all agreed to work together to achieve a larger goal. Each agent is self-interested, and by participating in the coalition it will receive a higher utility than if it did not participate in the coalition.

This concept is closer to congregating than team formation, since it gets at the notion that agents are self-interested and want to join a larger group to improve their rewards. However, coalition formation typically requires that the identities of the agents are known. Even if the actual problem of forming coalitions is distributed, there is common knowledge about who is involved. In congregating, an agent may not know the identities of other congregation members when it decides whether to join or not. In fact, the particular membership of a congregation may change over time. Consider the farmer's market example:

when a buyer of produce decides to go to the market, it does not know what other agents will be there, although it may have some predictions. The agents who are there may change from week to week, although it is likely that there will always be agents that fill particular roles, such as seller of tomatoes or buyer of flowers.

As we have suggested earlier, in many domains congregating could potentially be combined with coalition or team formation. Congregating would be used to subdivide a large, dynamic agent population into congregations. Coalition or team formation could then be applied within each congregation to allocate specific tasks. This would allow MASs to scale to larger numbers of agents than are practical for a single coalition formation algorithm.

9. Conclusions

This paper has presented congregating both as a metaphor for understanding how groups of agents can discover each other and as a formal model for describing the difficulty of particular multiagent coordinating problems. We have used the affinity group domain to apply CLRI to the congregating process. In doing so, we have shown that when agents have no means of describing themselves to each other, the problem complexity grows exponentially in the number of agents. By adding specialized agents known as labelers, the problem complexity reduces to linear in the number of labelers. If the number of labels is small, this is sufficient. However if the number of labels is large relative to the number of affinity groups, extra structure is needed to avoid exponential search. By allowing labelers to offer abstract labels arranged in a hierarchical structure, the congregating problem remains linear in the number of labels, which offers hope for the scalability of large multiagent systems.

We have also shown how congregating can be used to improve agents' net payoff in systems in which they must select another agent to coordinate with. By providing the agents with a set of loci at which they can congregate, they are able to self-select, thereby improving their average payoff. The congregating mechanism is essentially a self-organizing one; initially, agents that are happy in a congregation will stay there, providing a fixed location that other 'like-minded' agents can find. The only centralization that is required is that the agents all agree on the existence of a set of loci at the beginning of the system's lifetime. Once this initial commitment is made, agents need not consider the global state; they can simply concern themselves with the other agents in their congregation.

There are quite a few potential future directions for this research. One weakness of the current analytic approach, and CLRI more generally, is its reliance on PAC-learning assumptions, which do not allow for non-Markov system dynamics or domains with real-valued utility functions. Extending CLRI to model these sorts of domains is one current thread of our research.

This work has assumed that there is complete connectivity between loci, allowing agents to move from any locus to any other. The question of how these results would change if there was a topology over loci remains open. Additionally, this work does not explicitly treat nonstationarity in the consumer population. An important question to ask would be whether a system with multiple congregations is more or less stable as agents enter and leave.

Finally, we earlier stressed the notion of a congregation as an entity unto itself. This leads one to consider notions of group selection, whereby it becomes rational for agents within a congregation to behave altruistically, so as to increase the fitness of the rest of the congregation and improve their chances when competing with agents outside of the congregation. In environments in which agents typically interact within a congregation, but occasionally encounter agents from the population at large, this can be a very effective strategy (see [26] for ecological examples). This would also lead us to examine the relationship between individual and group-level learning: as individual agents develop more sophisticated strategies, how does the composition of the congregation change?

10. Acknowledgments

The authors thank Aaron Armstrong for his contributions to this research. Additionally, we thank our collaborators at the AI Lab at the University of Michigan, as well as the helpful comments made by the anonymous reviewers. This work was supported in part by the National Science Foundation under grant IIS-9872057.

References

1. Brooks, C. H. and E. H. Durfee: 1999, 'Congregation Formation in Information Economies'. In: *Proceedings of the AAAI-99 Workshop on AI in Electronic Commerce*. pp. 62–68.
2. Brooks, C. H., E. H. Durfee, and A. Armstrong: 2000, 'An Introduction to Congregating in Multiagent Systems'. In: *Proceedings of the Fourth International Conference on Multiagent Systems*. pp. 79–86.
3. Cohen, P. R. and H. J. Levesque: 1991, 'Teamwork'. *Nous* **25**, 487–512.
4. Davis, R. and R. G. Smith: 1983, 'Negotiation as a Metaphor for Distributed Problem Solving'. *Artificial Intelligence* **20**, 63–109.
5. Decker, K. S. and V. R. Lesser: 1997, 'Designing a Family of Coordination Algorithms'. In: M. Huhns and M. Singh (eds.): *Readings in Agents*. Morgan Kaufmann.

6. Durfee, E. H., D. L. Kiskis, and W. P. Birmingham: 1997, 'The Agent Architecture of the University of Michigan Digital Library'. *IEEE/British Computer Society Proceedings on Software Engineering* **144(1)**, 61–71. Special Issue on Intelligent Agents.
7. Gmytrasiewicz, P. J. and E. H. Durfee: 1993, 'Toward a Theory of Honesty and Trust Among Communicating Autonomous Agents'. *Group Decision and Negotiation (Special Issue on Distributed Artificial Intelligence)* **2**, 237–258.
8. Grosz, B. J. and S. Kraus: 1996, 'Collaborative plans for complex group action'. *Artificial Intelligence* **86**, 269–357.
9. Hu, J. and M. Wellman: 1998, 'Multiagent Reinforcement Learning: Theoretical Framework and an Algorithm'. In: *Proceedings of the Fifteenth International Conference on Machine Learning (ICML-98)*. pp. 242–250.
10. Jennings, N. R.: 1995, 'Controlling cooperative problem solving in industrial multi-agent systems using joint intentions'. *Artificial Intelligence* **75**, 195–240.
11. Ketchpel, S.: 1994, 'Forming Coalitions in the Face of Uncertain Rewards'. In: *Proceedings of the National Conference on Artificial Intelligence*. Seattle, WA, pp. 414–419.
12. Mas-Colell, A., M. D. Whinston, and J. R. Green: 1995, *Microeconomic Theory*. New York: Oxford University Press.
13. Mitchell, T. M.: 1997, *Machine Learning*. Boston, Massachusetts: WCB/McGraw-Hill.
14. Mullen, T. and M. P. Wellman: 1998, 'The Auction Manager: Market middleware for large-scale electronic commerce'. In: *Proceedings of Third USENIX Workshop on Electronic Commerce*.
15. Parunak, H. V. D.: 1987, *Distributed Artificial Intelligence*, Chapt. Manufacturing Experience With the Contract Net, pp. 285–310, Research Notes in Artificial Intelligence. Los Altos, CA: Morgan Kaufmann Publishers.
16. Sandholm, T. W. and V. R. Lesser: 1997, 'Coalitions among Computationally Bounded Agents'. *Artificial Intelligence* **94**(1), 99–137.
17. Schelling, T.: 1978, *Micromotives and Macrobehavior*. New York: W.W. Norton.
18. Sen, S., M. Sekaran, and J. Hale: 1994, 'Learning to Coordinate without Sharing Information'. In: *Proceedings of the National Conference on Artificial Intelligence*. pp. 426–431.
19. Shehory, O. and S. Kraus: 1998, 'Methods for Task Allocation via Agent Coalition Formation'. *Artificial Intelligence* **101**, 165–200.
20. Singh, M. P., A. S. Rao, and M. P. Georgeff: 1999, 'Formal Methods in DAI: Logic-based Representation and Reasoning'. In: G. Weiß (ed.): *Multiagent Systems: A Modern Aapproach to Distributed Artificial Intelligence*. Cambridge, MA: MIT Press, pp. 331–376.
21. Sycara, K., K. Decker, A. Pannu, M. Williamson, and D. Zeng: 1996, 'Distributed Intelligent Agents'. *IEEE Expert* **11**(6), 36–46.
22. Tambe, M.: 1997, 'Toward Flexible Teamwork'. *Journal of Artificial Intelligence Research* **7**, 83–124.
23. Vidal, J. M.: 1998, 'Computational Agents that Learn About Other Agents: Algorithms for Their Design and a Predictive Theory of Their Behavior'. Ph.D. thesis, University of Michigan.
24. Vidal, J. M. and E. H. Durfee: 1998, 'The Moving Target Function Problem in Multi-Agent Learning'. In: *Proceedings of the Third International Conference on Multi-Agent Systems*. Paris, France.
25. Vidal, J. M. and E. H. Durfee: 2002, 'Predicting the Expected Behavior of Agents that Learn About Agents: The CLRI Framework'. *Journal of Autonomous Agents and Multiagent Systems*. To appear.

26. Wilson, D. S.: 1980, *Natural Selection of Populations and Communities*. Menlo Park, California: Benjamin/Cummings.
27. Zlotkin, G. and J. S. Rosenschein: 1994, 'Coalition, Cryptography, and Stability: Mechanisms for Coalition Formation in Task Oriented Domains'. In: *Proceedings of the National Conference on Artificial Intelligence*. Seattle, WA, pp. 432–437.