The Pennsylvania State University
The Graduate School

THE EFFECT OF HETEROGENEOUS AGENTS IN SOCIO-COGNITIVE NETWORKS

A Thesis in
Computer Science and Engineering
by
Jui-Te Tseng

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Master of Science

December 2013

The thesis of Jui-Te Tseng was reviewed and approved* by the following:

Frank E. Ritter
Professor of Computer Science and Engineering
Thesis Advisor

Raj Acharya
Professor of Computer Science and Engineering
Department Head and Professor

Daniel Kifer
Assistant Professor of Computer Science and Engineering
Thesis committee

*Signatures are on file in the Graduate School.

Abstract

This research examined the effect of heterogenous agents in comparison with homogeneous agents in socio-cognitive network formation. I used cognitive agents connected to a virtual world and based on their interaction formed networks. The agents are different in movement strategies and starting locations. I found that: (a) the difference between homogeneous agents and heterogeneous agents, and (b) the starting locations of agents, both affect network formation. The results of heterogeneous agents in network formation is different from the average results of homogeneous agents and indicates heterogeneity changed the network formation fundamentally. So, network simulation should include heterogeneous agents.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

This research applied heterogeneous agents to social network formation simulations and considered several factors in social network formation, including spatiality, human memory, and heterogeneity of movement strategies and locations.

Agent-based modeling and simulation (ABMS) is a modeling paradigm [Macal and North, 2005] that is important in several reasons. First, by modeling, it gives us a way to reproduce the facts and understand them. Second, it gives us possibilities to predict large-scale properties that are hard to produce with real objects. Third, there are some circumstances that are hard to do experiments with only a single object. Fourth, it gives the possibilities to take the environment around the objects into consideration.

Using ABMS gives us an overall status of the system measurement. Using simulations, experimental factors can be examined thoroughly within an environment that can be controlled, repeat experiments as many times as wanted, and record the detailed data. There are several categories of agents in ABMS: cognitive agents, software agents, intelligent agents, and robotic agents [Fan and Yen, 2004]. This thesis used cognitive agents for the agents in the simulations.

Dunbar [1998] explained the relation between neocortex and relationships. He claimed that on average we have 150 friends memorized, and stable relationships require repeated memory activation. We can have connections between different groups and friends, but what we can remember daily is still limited. With Dunbar's

claim in mind, I built cognitive models and use them to form networks between agents. Capabilities and constraints of memory is thus important for this work.

The spatial factor is also what was examined in this research. In the simulations, agents move in a virtual world to meet other agents. The spatial factor is important because different agents have different movement strategies and this influences their network formation. I chose to change the starting locations of agents and found different starting location settings make difference.

Existing research of social agent models does not consider individual differences. Carley and Newell [1994] studied the nature of social agents. Fan and Yen [2004] studied modeling and simulating human teamwork behaviors. Sun [2009] studied cognitive architectures and multi-agent social simulation. Zhao et al. [2011, 2012a,b], and Kaulakis et al. [2012] studied socio-cognitive networks. All of these research do not consider heterogeneity.

Human variability is an important parameter that should be considered more often. It is suggested by Ritter and Norling [2006] that variability plays an important role in influencing task completion within a group. Due to their different properties and behaviors, heterogeneous agents seem to be the best tool to use to model and account for this variability in order to improve the accuracy of future network simulations. With agent-based simulation, we can explore the social network formation without making lots of assumptions in the beginning. We only need to define agent behaviors and environment constants then run experiments, read and interpret the results.

This thesis focuses on the effects of heterogeneity of agent models to social networks. Based on a National Research Council report [Zacharias et al., 2008], a more realistic model composition and models of groups and teams are suggested as a future direction. To explore heterogeneity of agents, I designed and ran experiments with: (1) different starting locations, and (2) different types of agents. I found the results with heterogeneous agents are different from networks with homogeneous agents, and the values of heterogeneous agents are not the average of the values from two homogeneous agents. Hence, running experiments with homogeneous agents is not enough and heterogeneous agents reflect the real cases

should be considered.

This thesis contains four additional chapters. Chapter 2 explains the background of this thesis, including agent based simulation, cognitive architectures, network analysis and VIPER, a simulation tool. Chapter 3 describes the environment of the experiments, the agent models I built for simulation, and the experiments I did. Chapter 4 presents the results and initial analyses them. Chapter 5 gives the conclusions of experiments, discusses the limitations of this work, and suggests some future work.

Chapter 2

Background

In this chapter, I present the background and related work of this thesis. In *Behavioral modeling and simulation: From individuals to societies*, Zacharias et al. [2008] listed several future directions in cognitive science and architecture development, and two of their suggestions are highly related to this thesis: enhance realism and models of groups and teams. From the aspect of enhancing realism, it is important to introduce variabilities characteristic of human behavior; from the aspect models of groups and teams, it means applying cognitive architectures to model group and teams is important [Mannix and Neale, 2005; Ritter and Norling, 2006]. With these two directions combined, it points out that homogeneous-agent simulations are not enough, and we need to consider simulations in a larger scale.

A simulation tool, VIPER (Virtual Implementation of Plural Environmental Representations) is suitable to simulate multiple agents in a spatial environment, and I want to use it to simulate network formations in a cognitive approach. The background of VIPER is discussed in section 4 of this chapter. Also, the related topics used in this thesis including (1) agent-based simulation, (2) cognitive architecture, and (3) network analysis, are reviewed in the following sections.

2.1 Agent-based Simulations

In Artificial Intelligence, the word "agent" is widely used, which can be defined as a software or hardware system that has the following properties: autonomy, social ability, reactivity, and pro-activeness [Wooldridge et al., 1995]. An agent

can continuously and automatically behave based on the environment, which may include other agents [Shoham, 1993]. Agent-based modeling can be applied to study numerous fields, such as business and organization, economics, infrastructure, crowds, social and culture, terrorism, military, biology [Macal and North, 2005].

A lot of work has been done in agent-based simulation; below are a list of some. The first related work is Schelling [1971] studied dynamic models of segregation. Other work includes Carley and Gasser [1999], who modeled organization theory; Carley and Newell [1994] discussed the Model Social Agent; Ritter et al. [2004] studied models related to behavior moderators. However, these studies do not have a large-scale simulation focused on networks.

We can group individual agents together to form a Multi-Agent System (MAS). MAS's form complex systems with independent agents interact with the environment and other agents. MAS's are required in some domains because the interaction between different subjects need to be handled [Stone and Veloso, 2000].

MAS's can be used to solve complex and unpredictable problems [Sycara, 1998] because they split the problem into subproblems and solve them in parallel. The characteristics of MAS's are (1) each agent has incomplete information for solving the problem (2) there is no global system control (3) data are decentralized (4) computation is asynchronous [Sycara, 1998]. The motivation to use MAS's includes: (1) the problem domain is too large to solve by a single agent (2) the way to solve the problem can be done by interactions between different components of the system (3) data need to be or is better to be distributed (4) some system performance can be improved, includes computational efficiency, reliability, extensibility, robustness, maintainability, responsiveness, flexibility, and reuse [Sycara, 1998].

Moreover, Stone and Veloso [2000] separated MAS's system into four categories based on agent variance and communication between agents: (1) homogeneous agents with no communication, (2) homogeneous agents with communication, (3) heterogeneous agents with no communication, and (4) heterogeneous agents with communication. In the second and fourth case, if agents can communicate seamlessly, we can regard the whole system as a single complex agent. In this thesis, the first and the third kind of cases are conducted. In other words, the agents in

this thesis have no communication with each other, and comparison between these two categories of agents sets that do not communicate are studied.

The related work of MAS's includes: Davidsson [2001] compared multi-agent-based simulation (MABS) with parallel and distributed discrete event simulation, object oriented simulation and dynamic micro simulation, and suggested some advantages of MABS, such as MABS is easy to simulate persons with different behavioral characteristics. Morgan et al. [2010] studied modeling of participation for small groups. As previous mentioned, this research differs because it focuses on a large-scale network formation simulation with heterogeneous agents.

2.2 Cognitive Architectures

A cognitive architecture comes from the concept of computer architecture, which defines the product of a design. Or I can borrow from Anderson's definition [Anderson, 2007, p.7]: "A cognitive architecture is a specification of the structure of the brain at a level of abstraction that explains how it achieves the function of the mind." In the following sections, I first gave an overview of cognitive architectures, then my choice of the architecture used in this thesis, ACT-R.

2.2.1 Overview of Cognitive Architectures

Anderson's definition points out the three parts of a cognitive architecture: brain (structure), mind (functional cognition), and the architectural abstractions between them. Some research has been done by ignoring one of these parts. If we ignore the brain, then it is classic information-processing psychology; If we ignore the mind, then it is eliminative connectionism; If we ignore the architecture, then it is rational analysis [Anderson, 2007]. Each area has its own progress and successfulness, but, as Newell's claim "you can't play 20-questions with Nature and win" [Newell, 1973], we definitely need a unified theory of cognition (UTC) because by this approach, the knowledge can be cumulated.

The ultimate goal of cognitive architectures is to produce a system that can generate human-level agents. We humans can have cognitive capacities on numerous tasks and seem to pursue an infinite variety of tasks. Therefore, we want a

human level agent that can do what a human mind does: interacting with a dynamic complex environment, planning, pursuing a variety of tasks, and learning.

Cognitive architectures are the fundamental of building cognitive agents because they provided basic cognition functions. Cognitive architectures are fixed (or slowly changing) and task independent, where the knowledge of agents grows by learning and is task focused. A cognitive architecture provides processes and memories that acquire, represent, and process knowledge of environment and tasks for reasoning, problem solving, and goal-oriented behavior [Laird, 2012].

Because of different long-term goals, we can separate research in cognitive architectures into three categories: (1) cognitive modeling: the goal of which is to provide models that match data from human behavior; (2) agent development: the goal of which is to develop agents that can interact with dynamic environments; (3) human-level agent development: the goal of which is to develop agents that have human cognition capabilities [Laird, 2012]. This thesis is related to the first and second categories.

2.2.2 ACT-R

ACT-R [Anderson et al., 2004] is a cognitive architecture that forms a specification of the structure of the brain at a level of abstraction and explains how it achieves the function of mind. It is built up by the data from psychological experiments as well as general assumptions of human cognition. It provides a framework and a DSL (Domain-Specific Language)[1] to let researchers develop their own domain-specific models which combined ACT-R assumptions with their own assumptions to the specific task. These assumptions can be tested by comparing the result and performance of a task between the model and humans. The model can thereafter use to predict performance in other tasks.

ACT-R has several components to fulfill the work.

Modules. ACT-R has two types of modules: the perceptual-motor module and the memory module. Perceptual-motor modules are interfaces to the real world, such as the visual module reflects human vision. The memory module includes two kinds of memory, one is declarative memory and the other is procedural mem-

[1]If we considered ACT-R is built in addition to Lisp as a library for modeling cognition.

ory. Declarative memory contains chunks, which are knowledge related to reality, such as the Pennsylvania State University is located in Pennsylvania. Procedural memory contains productions, which are knowledge about how we do things, such as how to hold a cup, and how to click a mouse.

Buffers. The model accesses modules through buffers. Buffers represent current state. A buffer can only holds one chunk at a time.

Pattern Matcher. At a given time, the pattern matcher searches among productions and finds one that matches the current state of the buffers. Only one production is fired at a time. A production can modify the buffers and change the state of the system.

Though it looks like a programming language, ACT-R is different from most other programming languages. Most programming languages have high-level operators executed on a low-level processor, and the commands are performed in a specific order. On the other hand, ACT-R is a low-level language that runs on a high-level cognition capacities processor. The actions depend on the production that matches the current state of buffers and modules.

In ACT-R, a chunk has an associated numerical value called its activation that indicates a chunk is useful in the current context. The chunk with the largest activation will be put into the retrieval buffer when a retrieval request is made. There is one constraint on the activation, called the retrieval threshold, which is the minimum activation value that a chunk can still be retrieved. The activation A_i can be computed by Equation 2.1:

$$A_i = B_i + \varepsilon \tag{2.1}$$

B_i is the base-level activation that reflects the frequency and recency of retrieval of a chunk. The equation of B_i is:

$$B_i = ln(\sum_{j=1}^{n} t_j^{-d}) \tag{2.2}$$

In Equation 2.2, n is the number of presentations for chunk i, t_j is the time since the jth presentation, and d is the decay parameter, usually set to 0.5. The equation represents that every time an item is presented, there is an increase in the base-level activation, but the effect of the increase decays as a power function

of the time since the presentation.

ε is the noise that contains two sources. The first source is a permanent noise associated with a chunk, and the second source is an instantaneous noise that will be recomputed while retrieving. **The noise value is generated according to a logistic distribution.**

The memory of cognitive models has the above properties, so when an agent does not meet its friend for a long time and the length of time is enough to lower the activation value of the chunk, it will forget the memory of that friend.

The models used in the experiments are discussed in the second section of Chapter 3, and some integration for connecting ACT-R models with the experiment environment is explained.

2.3 Network Analysis

Network analysis is a general topic related to numerous areas. The basic concept of network analysis is that nodes (or vertices) represent components of a network, and edges between nodes represent the connections between components. This can be applied to fields like biology, physics, economics, social studies, etc.

There are lots of work that has been done in this area, to name a few: as early as the 1970s, Zachary [1977] studied the opinion and information flow in small groups; Albert and Barabási [2002] studied statistical mechanics of complex networks; Kossinets and Watts [2006] studied social network in a large university by merging email interactions, personal attributes, and class information; Watts [1999] has a book about the small-world phenomenon. Newman [2003] has a great review on network analysis and structure.

Simulations are used to study network formation for this thesis. Through graph theory, the structure of network formation results is examined and compared. There are several network attributes based on graph theory in the network I want to examine and use them to compare between the experiment results:

- Edges: The edges in a graph formed from simulation is the first thing I want to examine because they represent the relationships between agents. I have a fixed number of agents run in the experiments, that make connections with

each other. Because of the memory assumption mentioned in the previous section, each agent may remember different numbers of friends, which means that nodes have different number of edges connected to them. Also, the edges are directed because an agent may remember another while the other does not.

- Assortativity (coefficient): Assortativity is a measure of homophily between nodes in a network based on some label or value assigned to a node, usually a node's degree. Homophily implies that we tend to be similar to our friends [Easley and Kleinberg, 2010]. The reason to use a node's degree to determine assortativity is because the connections between nodes of similar degrees are often found in networks. For example, in a social network, nodes with high degrees tend to connect with other high-degree nodes.

  Equation 2.3 is used to calculate assortativity [Newman, 2002]:

$$r = \frac{1}{\sigma_q^2} \sum_{jk} jk(e_{jk} - q_j q_k) \tag{2.3}$$

  $\sigma_q^2 = \sum_k k^2 q_k - [\sum_k k q_k]^2$ is the variance of the distribution q_k. q_k, the normalized distribution of the remaining degree, is $q_k = \frac{(k+1)p_{k+1}}{\sum_j j p_j}$. Here p_k is the probability of a randomly chose vertex with degree k. In such networks, the remaining degree, the number of edges leaving the vertex other than the one arrived, is considered. And e_{jk} is the joint probability distribution of the remaining degrees of the two end vertices of a randomly chosen edge. The properties of e_{jk} is $\sum_{jk} e_{jk} = 1$ and $\sum_j e_{jk} = q_k$.

  Assortativity is a value between 1 and –1. If the value equals 1, then it means the label between nodes have great assortative patterns; if the value equals 0, then the network is not assortative; if the value equals –1, then the network is disassortative. In analysis, I use degree of vertices to compute associativity.

- Global Cluster Coefficient: The cluster coefficient, or transitivity [Wasserman and Faust, 1994, p.243], is a measure that shows the nodes in a graph tend to cluster together. If the value is higher, then it means the nodes are grouped

with higher density. Global Cluster Coefficient is represented by the number of triplets between nodes divided by the maximum number of edges in the graph. A triplet means three nodes are connected to each other.

- Diameter: In a graph the distance between two vertices is the number of edges that connected them together in the shortest path. It is also known as geodesic distance. The diameter of a graph is the maximum geodesic distance in a graph.

- Reciprocity: Reciprocity measures the mutual connections between nodes in a directed network. Garlaschelli and Loffredo [2004] defined reciprocity as the correlation coefficient between the entries of the adjacency matrix of a directed graph, represented as Equation 2.4:

$$\rho \equiv \frac{\sum_{i \neq j}(a_{ij} - \bar{a})(a_{ji} - \bar{a})}{\sum_{i \neq j}(a_{ij} - \bar{a})^2} \tag{2.4}$$

  Where $a_{ij} = 1$ if there is a link from i to j, and $a_{ij} = 0$ if not. $\bar{a} \equiv \frac{\sum_{i \neq j} a_{ij}}{N(N-1)}$, N is the total number of edges in the graph. If it equals 1, then the network is a pure bidirectional one. If it equals 0, then it is a unidirectional one. With reciprocity, we can understand the probability of mutual direction links in a network.

These attributes are used to analyze the network formation results of the experiments in Chapter 4.

2.4 VIPER

To conduct the simulation, a simulation environment is required. The choice for this thesis is VIPER [Hiam et al., 2011; Zhao et al., 2012a], a text-based, multi-agent simulation environment that is built on top of NakedMud [Hollis, 2009], a content-less MUD engine. A MUD (Multi-User Domain) is a text-based multi-player real-time virtual world with role-playing games as the main content. Typically, players connect to the server and send commands to interact with other

objects in the virtual world. A player can read a description of rooms, objects, players, and non-player characters and perform actions.

VIPER takes advantages of this server-client architecture to run experiments. The architecture ensures that each client runs independently without interference and the server can handle the overall conditions and ensure clients start to run at the same time. The server and clients communicate through the Telnet protocol with text messages, which is a common protocol of MUDs.

As a usual MUD system, my cognitive agents act as a player, read messages, and behave based on the cognitive model inside a 2-D room-based map. The map used by VIPER is configurable with different spatial settings and can be chosen during the preparation stage of experiments. Basically, each room has doors in four directions: north, south, east, west. If a room is at the edge of the map, then the side is a wall. An agent is placed into a particular room after being logged into the server. It can then take actions with other objects inside the same room or decide to move to other adjacent rooms.

The basic actions that an agent can do inside VIPER includes: (1) look: look means look around the room it stays in, which means VIPER will give the agent information about this room, includes who else are also in this room and the information of other objects. (2) say: an agent can say some words and they will be heard by other agents in the same room. (3) north: go to the room that located in the north side of this room. (4) south: go to the room that located in the south side of this room. (5) east: go to the room that located in the east side of this room. (6) west: go to the room that located in the north side of this room.

The reasons VIPER is used as the experiment environment is because:
(1) we can have essentially unlimited agents if we have enough powerful hardware. Due to the server-client architecture, we can run them on different machines to help performance. (2) VIPER provides a great logging system to log the behavior of agents both from the view of agent itself or from a system-wide perspective. With both logging results, it is easy to examine the details of network formation.

Zhao et al. [2011, 2012a,b], and Kaulakis et al. [2012] have done different experiments based on VIPER with different factors, such as environment configuration, agent size, or cognitive models. But none of them has run experiments with heterogeneous agents, which is what this study is concerned with.

2.5 Summary

This chapter reviewed (1) agent-based simulations, includes MASs and heterogeneous agents in MASs. (2) cognitive architectures and my choice of cognitive architecture: ACT-R. (3) Network analysis for the results of network formations. (4) VIPER, the simulation environment for the experiments.

A more specific experiment environment is explained in the first section of Chapter 3 on the experiment method, which gives the server specification and software versions that is used in the experiments.

Chapter 3

Experiments

To test the assumption that the starting location of an agent and heterogeneous agents influence network formation, VIPER is used with the cognitive agents to run two experiments of network formation simulation and take measurements. This chapter describes about the setup of the experiment environment. The results and analysis are presented in the next chapter.

3.1 Computing Environment of the Experiments

The experiments have the following setup: the VIPER server is installed on a machine with the specification shown in Table 3.1, running Mac OS X 10.5.8 with Python 2.5.1. VIPER itself is built up with NakedMud version 3.8.1. The agents ran on a machine with the specification shown in Table 3.2, running RedHat Linux with Python 2.6.6, SBCL 1.0.58, and ACT-R 6.0.

Though the specification of the machines is listed here, it does not mean the experiments need to be run on the same specification, but the software version may need to be verified or modified.

A Python script was written for reusing and convenience. It follows the following procedures to conduct an experiment.

Administrator connection. The script starts another Python script that connected to the VIPER server as an administrator. After being connected, the administrator user waits for all agents to connect to the server. The administrator then moves the agents to the starting locations specified by the command line

Table 3.1. The hardware specification of the VIPER server.

Attribute name	Value
Model Name	Power Mac G5
Model Identifier	PowerMac7,2
Processor Name	PowerPC 970 (2.2)
Processor Speed	2 GHz
Number Of CPUs	2
L2 Cache (per CPU)	512 KB
Memory	1 GB
Bus Speed	1 GHz

Table 3.2. The hardware specification of the server running clients.

Attribute name	Value
CPU	Intel Dual Core i5 2400 Processor (3.1GHz, 6M)
RAM	8GB DDR3 Non-ECC SDRAM, 1333MHz
Hard disk	250GB 7,200 RPM 3.5" SATA, 6.0Gb/s Hard Drive with 8MB Cache

argument. It also moves other characters that are not connected during this experiment to a different map so they will not influence the current experiment. It then sends out the logging command to start a new log, and sends out the "start simulation" broadcast messages to notify all the agents to start their actions, which also means the initiation of an experiment.

Agents connection. The agents connect to the server one-by-one to avoid the maximum simultaneous connection limit on the server. After connected to the server, the agents wait for the administrator to send out the "start simulation" broadcast messages as a notification to start actions based on their model. They will continue to take actions until reaching the running time limit. They will then record their memory status in a local CSV file and disconnect from the server.

Logging and clean up. After the set length of time, the script waits for the agents processes to stop. If all the agents stop properly, then it is a successful experiment and the script will copy the system-wide log file from the server and collect the memory result of agents into a combined file for analysis. If some agent exits exceptionally, the Python script will terminate other processes, make error logs, and restart a new experiment run.

The map configuration I used on the VIPER is all the same in each run, which is a full 5 x 5 grid with grid ratio 1.0, also know as well connected grid in previous work [Zhao et al., 2012a], as shown in Figure 3.1.

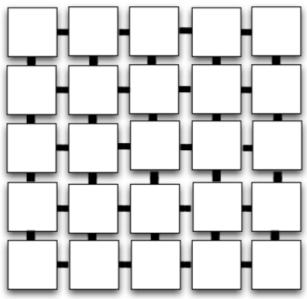

Figure 3.1. The 5 x 5 grid that is used in the experiments.

3.2 Agent Models in the Experiments

The client side of the experiment is cognitive agents based on ACT-R, a unified theory of cognition and a cognitive architecture [Anderson et al., 2004]. As a cognitive architecture, it reflects assumption of human cognition and provides the memory module of humans that I want to examine.

A library—TAWA (Telnet Agent Wrapper for ACT-R)—is used to connect ACT-R agents to VIPER, which is a Telnet-based service. It adds features like synchronization, logging, halting, and writing results to CSV files. The synchronization function is especially important because it ensures that all the agents start their actions at the same time, which is required for the experiments. With TAWA, an agent can perform the following actions: (1) send: send a message to VIPER. This is the way that agents send commands and take actions in the MUD system. (2) say: An agent says a given string. (3) list-exits: the system will give a list of strings contains the names of exits of a room. (4) list-peers: the system will give a list of strings with the names of other characters in the room. (5) list-front: the system will give a list of strings with the front-matter that describe the room itself. (6) look: the system give the description of current room.

The following chunk types are designed in the ACT-R models: (1) friend-list: a list of friends the model remembers. (2) friend: the chunk type memorized in each

agent's friend-list, including the name and the id of a friend. (3) event: includes information about the event that met a person in a particular location as well as the action. (4) walk-info: the current state. (5) task: contains information of current subgoal that agents going to do during the simulation. (6) route: contains the information of a route when the agent is in a particular room. (7) counter: the agent walks based on the information of this counter. On the other hand, the model has the following production rules sets: (1) moving strategy, and (2) rules when meet other agents.

I have two models that have different moving strategies. The first model moves randomly around rooms, so I named it "random-walk" model. When it goes into a room, it reads the information from the server, knows what are the exits of this room, randomly chooses one direction, and goes to the next room.

When an agent is in a room, it will look around and see if there is any other agent in the room. If there is any other agent, it will first search its memory to see if it has the memory of this agent, and retrieve the information from memory and strengthen it. If the other agent is new, it will create a new memory chunk related to this agent, and remember this agent as its friend. It will continue to memorize all the agents in the same room.

The other model shared the same behavior of memorizing friends, but has a different moving strategy. When it starts to move during the experiment, it will randomly choose one possible exit of the starting location, but later it will walk in the same direction across the next room, turn left, and walk in the same direction across a room, and repeat this behavior through the grid. **In other words, it tries to walk surrounding the walls of a room in a counter-clockwise movement.** This means it tends to stick to a place and does not go too far away from the starting location. Also, because all the agents move with a fixed speed, the overlap between agents is limited if they follow the same path. This model is named "round-walk" model. Figure 3.2 shows an example of the path.

In the simulation, when two agents meet each other, they memorize the other agent as their friend. This is implemented by the ACT-R memory chunks mentioned previously. Every memory chunk in ACT-R has its activation value and threshold. The memory need to keep activate or the activation value will decay over time. So if two agents do not meet each other again before the memory decays

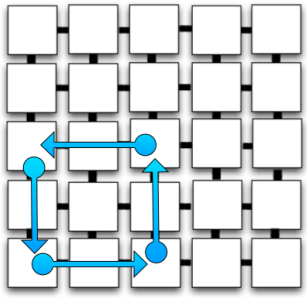

Figure 3.2. An example of round-walk agent's movement strategy.

down to the threshold, they will forget the other agent as their friend. If there are multiple agents in the room, the agent will remember them all as friends.

3.3 Explanation of Experiment 1

In the first experiment, I examine the effect of different starting locations in network formation. VIPER is used as the experiment environment and the map is the well-connected grid explained in Section 3.1. Following the common setting of experiments, this type of experiment uses the attributes shown in Table 3.3.

Table 3.3. The attributes of experiment 1.

Attribute	Value
Running time	300 seconds
Number of agents	30
Map	5 by 5 grid
Agent type	random-walk agent

The running time is chosen based on previous work [Zhao et al., 2012a], where after 300 seconds, the number of links formed is relatively stable. There is a delay of 16 seconds that after every movement, and also a 4 seconds delay after other actions.

There are four types of starting locations: (1) center: all the agents start in the center of the map. (2) corner: all the agents start from a corner of the map. (3) random: the agents are assigned their starting location randomly with replacement. (4) equally distributed: the agents are assigned their starting location equally distributed across all the rooms.

All four settings were run 10 times. **The number of runs is chosen based on previous work [Kaulakis et al., 2012].** I recorded the memory and the behaviors of the agents.

3.4 Explanation of Experiment 2

In the second experiment, I wanted to examine the effect of heterogeneous agents in the network formation simulation in addition to the first experiment. The same as Experiment 1, VIPER is used as the experiment environment and the map is the well-connected grid explained in section 3.1. Following the common setting of experiments, this experiment uses the attributes shown in Table 3.4.

Table 3.4. The attributes of experiment 2.

Attribute	Value
Running time	300 seconds
Number of agents	30
Map	5 by 5 grid

There are three different sets of agents: (1) All agents are random-walk agents. (2) All agents are round-walk agents. (3) Half of the agents are random-walk agents, and half of the agents are round-walk agents. There are also four types of starting locations as Experiment 1: (1) center: all the agents start from the center of the map. (2) corner: all the agents start from a corner of the map. (3) random: the agents are assigned their starting location randomly. (4) equally distributed: the agents are assigned their starting location equally distributed across all the rooms.

Based on these factors, I have 12 conditions, and I ran each one 10 times. I recorded the memory and the behaviors of the agents and the result is explained in section 4.2.

3.5 Summary

This chapter described the details of the experiment environment, agent models used in the experiments, and the factors in the experiments that were run. The

next chapter describes the results of experiments and provides a summary of the results.

Chapter 4

Results and Analyses

This chapter presents the results and analysis of experiments. It is presented as two sections: the results of Experiment 1 and the results of Experiment 2.

There are two types of data generated by the simulation. One is log data extracted from VIPER, which recorded the actions that every agent took during the simulation as well as the real time of events. The other is egocentric data that represents an agent's declarative memory, which records the friends that an agent remembers at the end of the simulation. This gives me details about the memory results of the cognitive model.

In the following sections, egocentric data is used to form the network for analysis. From the data, an agent move on the map approximate 7 times. In such data, every agent is represented as a node. If an agent A has a memory record of another agent B, there is a directed link from A to B. **In this research, the activation values of memory are considered. Zhao et al. [2013] found that having threshold of the activation values changes the total number of links in a network. With perfect memory, the total number of edges formed in a network is larger the condition with memory activation threshold.**

For this research, the threshold of activation values of the memory is set to –3.5, which means that an agent forgets the relationship if the activation value is lower than –3.5. The edges are formed only when the activation value related to an edge is greater than the threshold. So in the analysis, 3.5 is added to the original activation value and the link is

weighted with the modified value. Therefore, a directed weighted graph is formed with nodes represent agents and the weighted links between nodes represent their friendships. Figure 4.1 is an example of a direct weighted graph.

Box plots are used to represent the results. The thick line inside the box is the median, and the bottom and the top of the box are the first and third quartiles. The top line and the bottom line are the maximum and minimum of the data, and outliers are represented as small circles.

To indicate the significance between each data set, I made two-sample t-tests to compute the significance of the differences between the two data sets. Equation 4.1 is used to calculate the t value.

$$t = \frac{\bar{x_1} - \bar{x_2}}{\sqrt{\frac{SD_1^2}{n_1} + \frac{SD_2^2}{n_2}}} \tag{4.1}$$

$\bar{x_1}$ and $\bar{x_2}$ are the average of both samples, SD_1 and SD_2 are standard deviations of both averages, and n_1 and n_2 are the number of samples. Based on t-table with $N = 10$, $\alpha = 0.05$, and 2-tailed, I emphasized values greater than 2.262 with a bolded font because the two data sets have data difference.

4.1 Results of Experiment 1

In Experiment 1, the random-walk agent model is used with different starting locations. As previous chapter mentioned, there are four possible starting locations: center, corner, randomly distributed, and equally distributed. After the graph formation, I compare the properties of the graph. I give descriptive statistics first, and at the end give the inferential statistics as a summary.

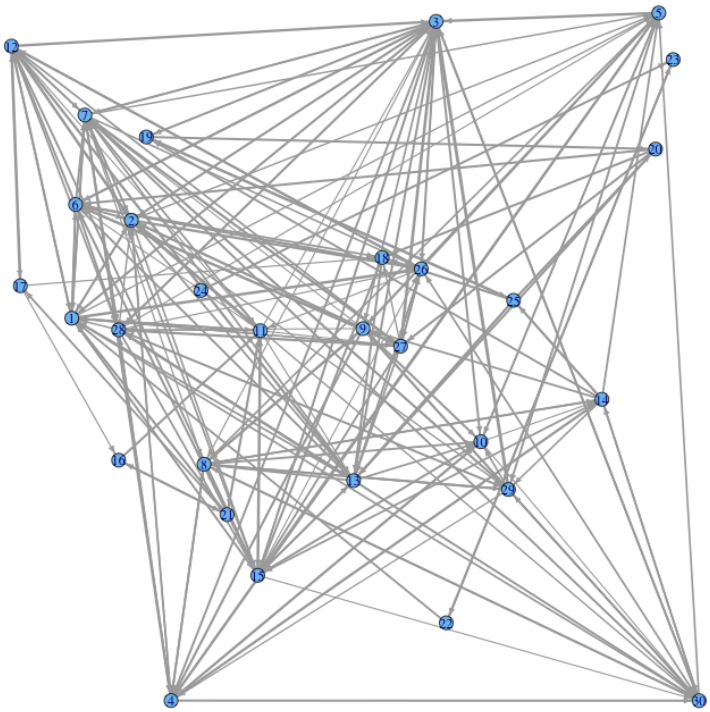

Figure 4.1. An example of the network formation result.

Edges. As mentioned in Chapter 2, edges between nodes means the relationships between agents, and they are the first thing to examine. From the cumulative data, it shows an overview of the network formation. **The maximum count of edges is** $30 \times 29 = 870$ **(bi-directional links).**

Figure 4.1 shows that the starting locations influenced the number of edges formed during simulation. The center starting location and the corner starting location are two starting locations where agents all started in the same room. As a result, these two types of starting locations encouraged agents to form more relationships. On the other hand, randomly and equally distributed locations means agents were put all around the grid, so it is harder for them to meet other agents.

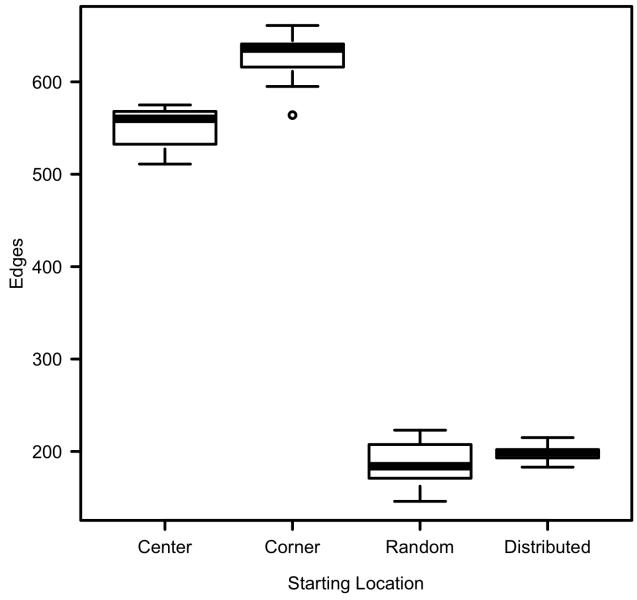

Figure 4.2. The box plot of total edge count with four different starting locations (N = 10 runs per box). Maximum is 870.

Table 4.1. t values of the edges between starting location pairs with N = 10. It is a two-tailed test with degree of freedom = 19. The bold values are t value >2.26 or p<.05.

Starting location pair	t value
Center/Corner	**7.15**
Center/Random	**34.99**
Center/Distributed	**47.32**
Corner/Random	**39.01**
Corner/Distributed	**49.57**
Random/Distributed	1.30

Assortativity. Associativity of nodes is used to see the homophily between agents, where homophily implies that nodes tend to be similar to their connected nodes. As mentioned in Chapter 2, the degree of nodes is used to compute assortativity. From Figure 4.3, the box plot of the results, the data sets with the center starting location and the corner starting location have disassortativity. On the other hand, the data sets with randomly distributed starting locations and equally distributed starting locations have assortativity and there is no significant difference between these two data sets (with t value = 1.255). This again explained

that the starting location is a factor that influenced network formation, and randomly distributed starting locations and equally distributed starting locations are similar.

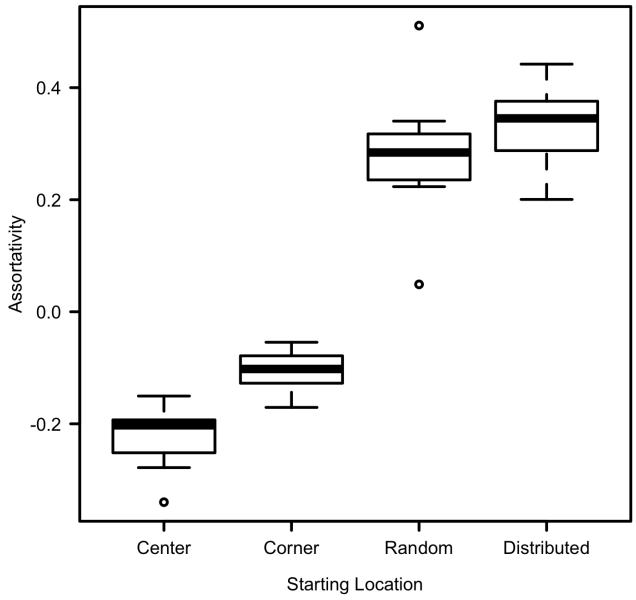

Figure 4.3. The box plot of associativity with four different starting locations (N = 10 runs per box).

Table 4.2. t values of the assortativity between starting location pairs with N = 10. It is a two-tailed test with degree of freedom = 19. The bold values are t value >2.26 or p<.05.

Starting location pair	t value
Center/Corner	**6.05**
Center/Random	**13.51**
Center/Distributed	**19.84**
Corner/Random	**10.86**
Corner/Distributed	**17.15**
Random/Distributed	1.26

Global cluster coefficient. In graph theory, a clustering coefficient indicates how much the nodes in a graph cluster together. The global cluster coefficient gives the global indication of such properties. Figure 4.4 shows that when the data sets are with the center starting location or the corner starting location, the network is highly clustered together because the global cluster coefficient is around 1.0. On the other hand, the global cluster coefficient with the random distributed starting location and equally distributed starting location are around 0.7, which is less clustered than the center starting location and the corner starting location. The t value between the center and corner starting location is 2.498, and the t value between randomly distributed and equally distributed starting locations is 0.786. Both t values show strong relation between two data sets.

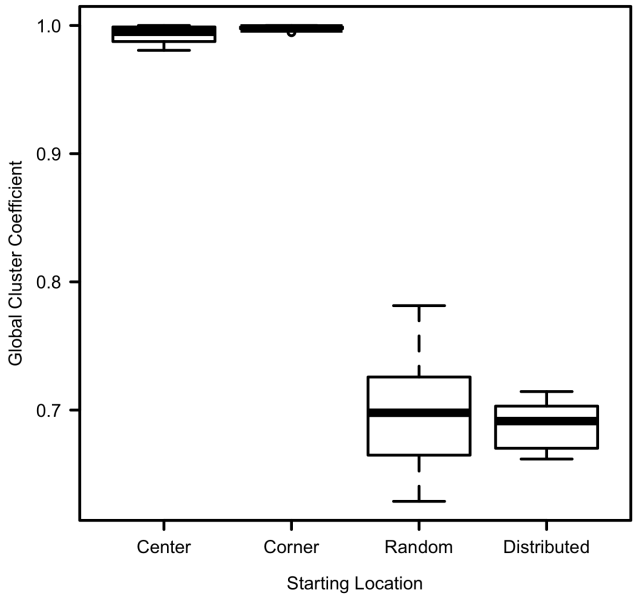

Figure 4.4. The box plot of global cluster coefficient with four different starting locations (N = 10 runs per box).

Table 4.3. t values of the global cluster coefficient between starting location pairs with N = 10. It is a two-tailed test with degree of freedom = 19. The bold values are t value >2.26 or p<.05.

Starting location pair	t value
Center/Corner	**2.50**
Center/Random	**20.59**
Center/Distributed	**49.84**
Corner/Random	**21.21**
Corner/Distributed	**53.99**
Random/Distributed	0.79

Diameter. The diameter of a graph is the longest path length of the graph. Figure 4.5 shows that the diameter with the center and the corner starting location share similar numbers, which are around 5, and the t value of this pair is 0.526. On the other hand, the random and the equally distributed starting location have similar diameter numbers, which are about 12, and the t value of this pair is 0.058. This measurement also fit my other measurements that the agents tend to cluster together when the starting location is the center or the corner.

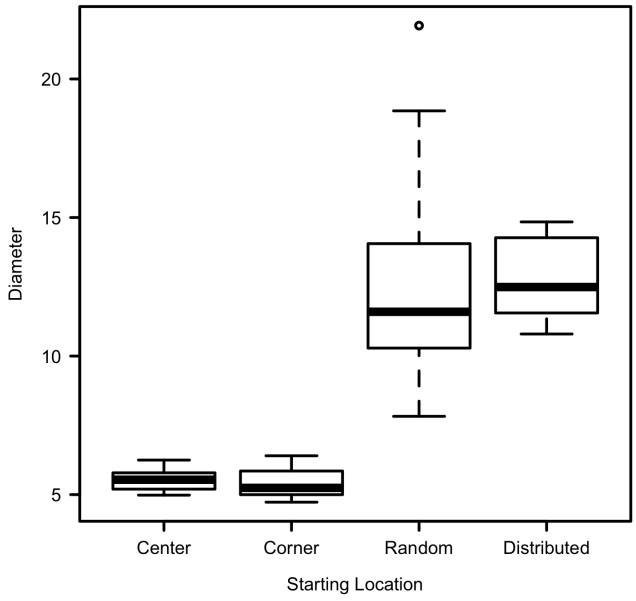

Figure 4.5. The box plot of diameter with four different starting locations (N = 10 runs per box).

Table 4.4. t values of the diameter between starting location pairs with N = 10. It is a two-tailed test with degree of freedom = 19. The bold values are t value >2.26 or p<.05.

Starting location pair	t value
Center/Corner	0.53
Center/Random	**5.61**
Center/Distributed	**15.24**
Corner/Random	**5.68**
Corner/Distributed	**15.05**
Random/Distributed	0.06

Reciprocity. Recall that reciprocity is the probability of the mutual connections between nodes in a directed network. Figure 4.6 shows that this attribute is different from other attributes with only the corner starting location has difference than other starting locations, and other starting locations do not significant difference.

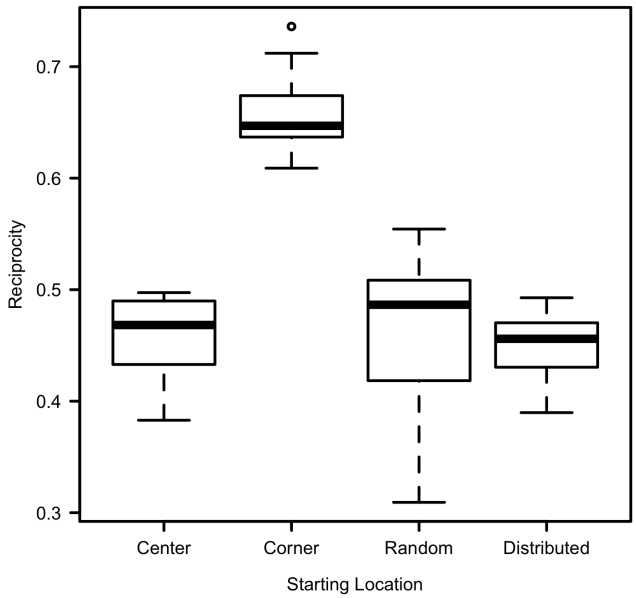

Figure 4.6. The box plot of reciprocity with four different starting locations (N = 10 runs per box).

Table 4.5. t values of the reciprocity between starting location pairs with N = 10. It is a two-tailed test with degree of freedom = 19. The bold values are t value >2.26 or p<.05.

Starting location pair	t value
Center/Corner	**12.33**
Center/Random	0.15
Center/Distributed	0.50
Corner/Random	**8.17**
Corner/Distributed	**13.94**
Random/Distributed	0.48

Significant difference between different factors. Table 4.1 is the t-test values between starting location pairs.

It presents that the center and corner starting location do not have a significant difference when considering global cluster coefficient and diameter. Also, randomly distributed starting locations and equally distributed starting locations do not have a significant difference for all the properties I examined. However, reciprocity is not a property that can separate between data sets well, because half of the data pairs do not have a significant difference.

4.2 Results of Experiment 2

Experiment 2 extended the setting of Experiment 1 with heterogeneous agents. As previously noted, I have two types of agent models, so I have three different types of agent sets: (1) all the agents are random-walk agents. (2) all the agents are round-walk agents. (3) Half of the agents are random-walk agents and half of the agents are round-walk agents. After the graph formation, I compared the properties of the measures.

The abbreviations I used in the graphs in this section have the meanings as in Table 4.2 and 4.3. Again, I give descriptive statistics first, and at the end give the inferential statistics as a summary.

Table 4.6. Abbreviations used in graphs.

Abbreviation	Meaning
C	starting location: Center
E	starting location: corner (Edge)
R	starting location: Randomly distributed
D	starting location: equally Distributed
(Ran)	agent model: Random-walk agent
(Rd)	agent model: Round-walk agent
(HH)	Half are random-walk agents and Half are round-walk agents

Edges. Figure 4.7 shows that starting locations had a greater influence than the types of agents. When the starting location is randomly distributed or equally distributed, even the difference between random-walk agents and round-walk agents is not significant, so heterogeneity does not affect much. But when the starting location is center or corner, there is the difference between random-walk agents and round-walk agents, and heterogeneity agents also have formed different number of edges. There is only no significant difference between round-walk agents and heterogeneous agents with the center starting locations.

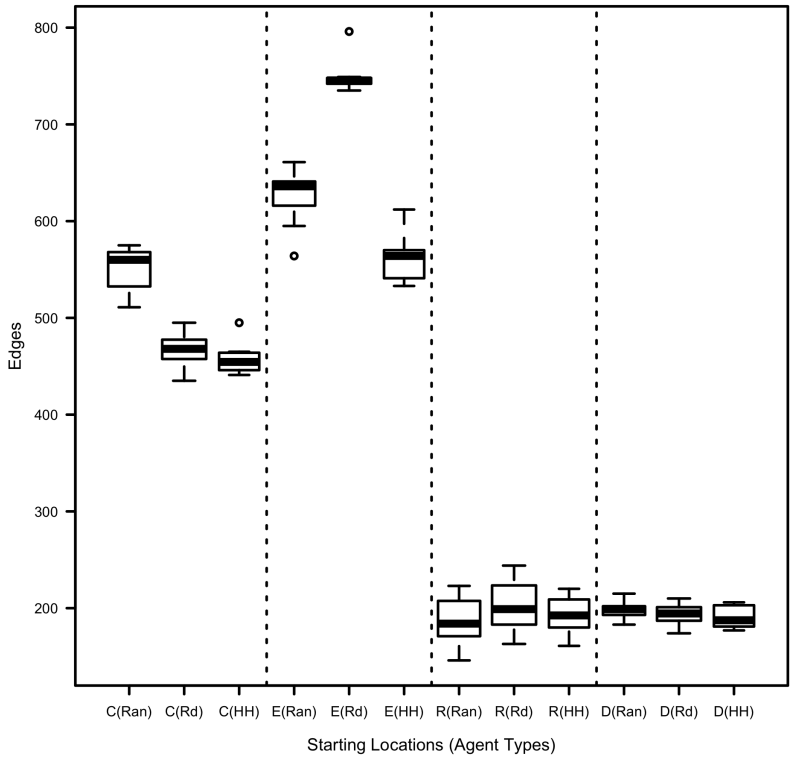

Figure 4.7. The box plot of total edge count for Experiment 2. N = 10 runs per box. Maximum is 870. (Labels from Table 4.2)

Table 4.3 is the t values of the edges between agent set pairs. It shows that when the starting locations are randomly or equally distributed, the difference between agent models or heterogeneity does not influence the edges of a network.

Table 4.7. t values of the edges between agent set pairs of all starting location settings. N = 10. It is a two-tailed test with degree of freedom = 19. The bold values are t value >2.26 or p<.05.

Starting location	Agent set pairs	t values
Center	Random/Round	**9.66**
Center	Random/Hetero	**10.31**
Center	Round/Hetero	0.99
Corner	Random/Round	**14.18**
Corner	Random/Hetero	**7.21**
Corner	Round/Hetero	**19.57**
Randomly distributed	Random/Round	0.69
Randomly distributed	Random/Hetero	0.23
Randomly distributed	Round/Hetero	0.51
Equally distributed	Random/Round	1.04
Equally distributed	Random/Hetero	1.78
Equally distributed	Round/Hetero	0.72

Assortativity. Figure 4.8 is the box plot of the results. The the center starting location and the corner starting location conditions have disassortativity and the data sets with random starting locations and equally distributed starting locations have assortativity. There is an exception, when running round-walk agents with the corner starting location, the data have variance that is possible to form assortativity. The reason is because the moving strategy of round-walk agents, they sometimes stuck around the edges of the grids. But considered heterogeneity, I found that with the center starting location, the corner starting location, and randomly distributed location, heterogenous agent set has the mean value between two homogeneous sets. But when the starting location is randomly distributed, heterogeneous agents has less associativity than other two types of homogeneous agent sets and the value of associativity is near 0.

Table 4.4 is t values of the assortativity between agent set pairs. It shows that heterogeneity influences assortativity in most of the cases.

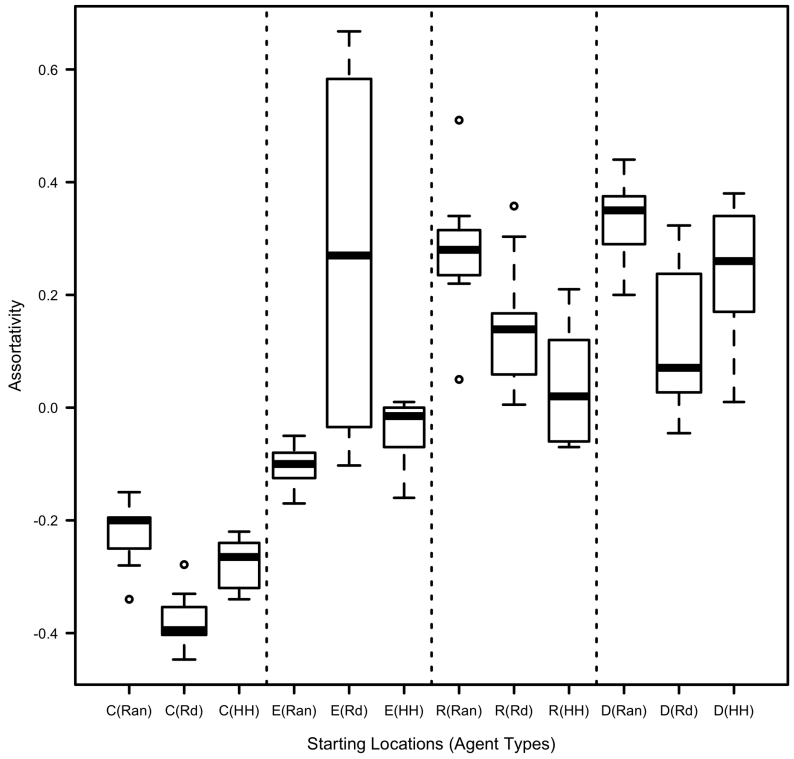

Figure 4.8. The box plot of associativity for Experiment 2. N = 10 runs per box. (Labels from Table 4.2)

Table 4.8. t values of the assortativity between agent set pairs of all starting location settings. N = 10. It is a two-tailed test with degree of freedom = 19. The bold values are t value >2.26 or p<.05.

Starting location	Agent set pairs	t values
Center	Random/Round	**10.69**
Center	Random/Hetero	**3.46**
Center	Round/Hetero	**6.33**
Corner	Random/Round	**2.69**
Corner	Random/Hetero	**3.35**
Corner	Round/Hetero	1.99
Randomly distributed	Random/Round	**4.15**
Randomly distributed	Random/Hetero	**5.01**
Randomly distributed	Round/Hetero	0.85
Equally distributed	Random/Round	**6.38**
Equally distributed	Random/Hetero	**2.42**
Equally distributed	Round/Hetero	**2.95**

Global cluster coefficient. Experiment 1 shows that when the starting location is at the center or corner, the agents are highly clustered together because the global cluster coefficient is around 1.0. In Figure 4.9, it also reproduce the result and the difference between homogeneous agents and heterogeneous agents is less than 0.25. Also, though the values of global cluster coefficient with random-walk agents and round-walk agents are both around 1, heterogeneous agents has lower values. This shows heterogeneous agents changes the network formation on this property.

On the other hand, when the starting location is randomly distributed, heterogeneous agents has lower values; when the starting location is equally distributed: heterogeneous agents have the value between two homogeneous agents. The values shows that heterogeneous agents have different network formation based on global cluster coefficient. Table 4.5 presents the t values of the global cluster coefficient between agent set pairs.

Table 4.9. t values of the global cluster coefficient between agent set pairs of all starting location settings. N = 10. It is a two-tailed test with degree of freedom = 19. The bold values are t value >2.26 or p<.05.

Starting location	Agent set pairs	t values
Center	Random/Round	**2.70**
Center	Random/Hetero	**6.87**
Center	Round/Hetero	**2.55**
Corner	Random/Round	1
Corner	Random/Hetero	**3.87**
Corner	Round/Hetero	**3.87**
Randomly distributed	Random/Round	**6.48**
Randomly distributed	Random/Hetero	**8.14**
Randomly distributed	Round/Hetero	0.08
Equally distributed	Random/Round	**16.15**
Equally distributed	Random/Hetero	**3.37**
Equally distributed	Round/Hetero	**9.77**

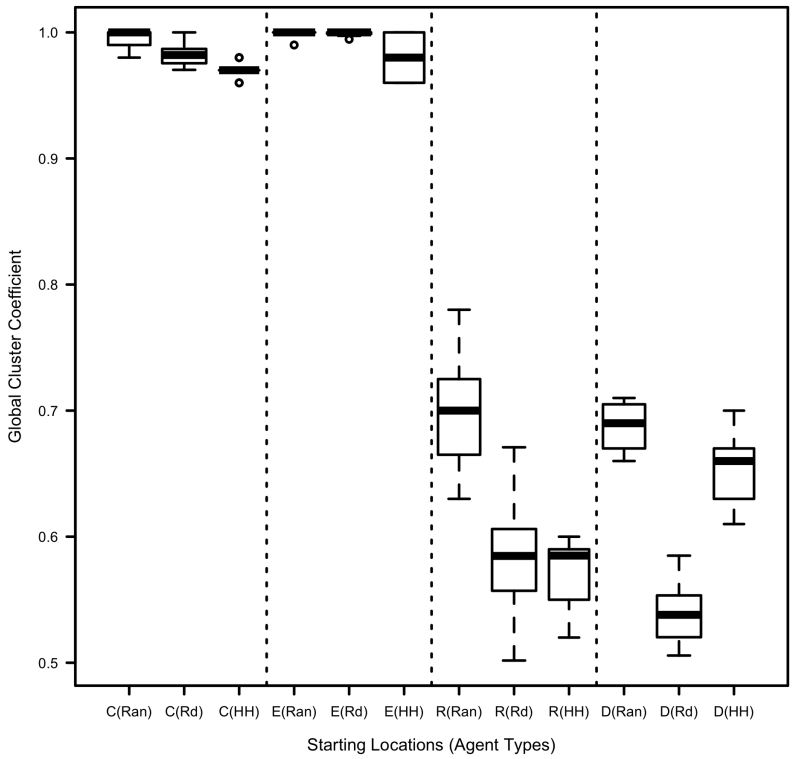

Figure 4.9. The box plot of global cluster coefficient for Experiment 2. N = 10 runs per box. (Labels from Table 4.2)

Diameter. Experiment 1 shows that with the center or the corner starting location, the diameter of the network formation graphs is smaller. The result in Figure 4.10 follows this analysis, and the heterogeneous agents also moderate the network formation. With the center starting location and the equally distributed starting locations, heterogeneous agents have the value between homogeneous agents. But with the corner starting location, heterogeneous agents have higher diameter; with the randomly distributed starting location, heterogeneous agents have lower diameter. Table 4.6 is t values of the diameter between agent set pairs.

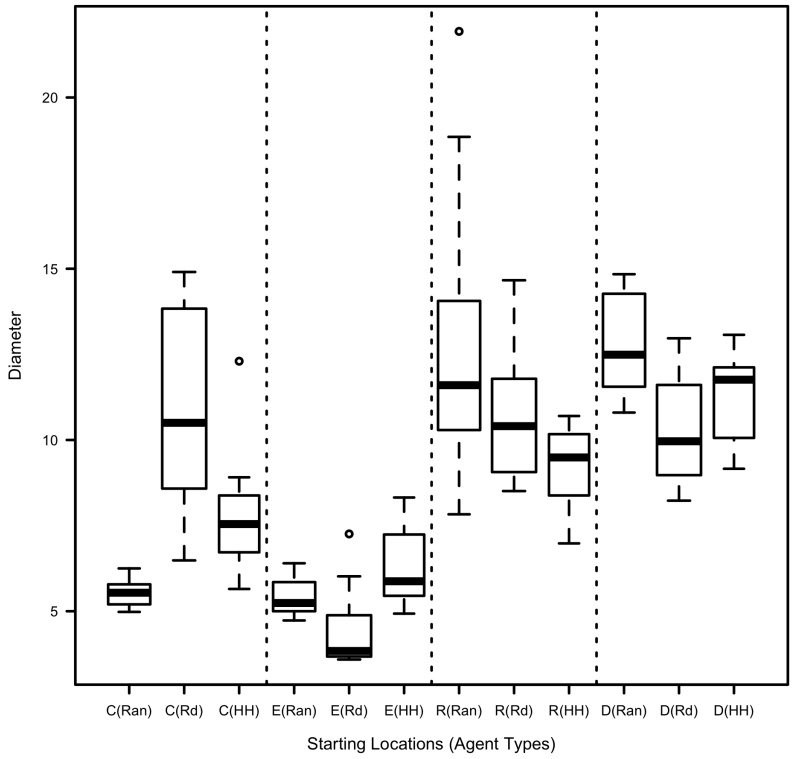

Figure 4.10. The box plot of diameter for Experiment 2. N = 10 runs per box. (Labels from Table 4.2)

Table 4.10. t values of the diameter between agent set pairs of all starting location settings. N = 10. It is a two-tailed test with degree of freedom = 19. The bold values are t value >2.26 or p<.05.

Starting location	Agent set pairs	t values
Center	Random/Round	**3.70**
Center	Random/Hetero	**4.11**
Center	Round/Hetero	1.52
Corner	Random/Round	**3.43**
Corner	Random/Hetero	2.09
Corner	Round/Hetero	**4.45**
Randomly distributed	Random/Round	1.73
Randomly distributed	Random/Hetero	**2.45**
Randomly distributed	Round/Hetero	1.26
Equally distributed	Random/Round	**6.22**
Equally distributed	Random/Hetero	**2.67**
Equally distributed	Round/Hetero	**3.75**

Reciprocity. Recalled that reciprocity is the probability of the mutual connections between nodes in a directed network. Figure 4.11 shows that this attribute is different from other attributes, where only the corner starting location has difference than other starting locations. The reciprocity with round-walk agents and the corner starting location is near 1.0, but when using heterogeneous agents, the value is more near to the results with only random-walk agents. Table 4.7 is t values of the reciprocity between agent set pairs.

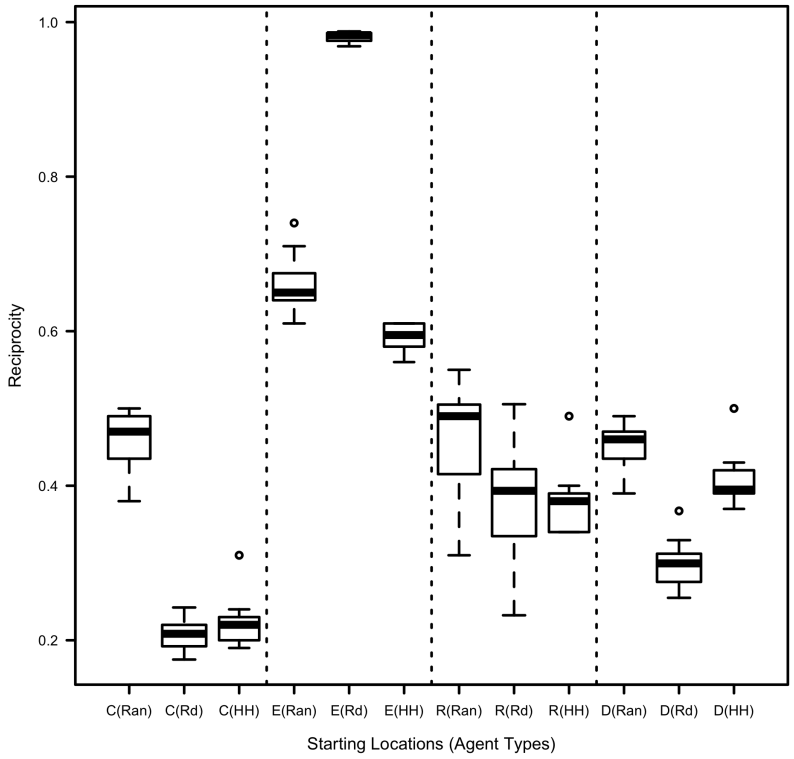

Figure 4.11. The box plot of reciprocity for Experiment 2. N = 10 runs per box. (Labels from Table 4.2)

Table 4.11. t values of the reciprocity between agent set pairs of all starting location settings. N = 10. It is a two-tailed test with degree of freedom = 19. The bold values are t value >2.26 or p<.05.

Starting location	Agent set pairs	t values
Center	Random/Round	**18.37**
Center	Random/Hetero	**16.10**
Center	Round/Hetero	1.08
Corner	Random/Round	**22.64**
Corner	Random/Hetero	**4.96**
Corner	Round/Hetero	**52.04**
Randomly distributed	Random/Round	**2.93**
Randomly distributed	Random/Hetero	**2.72**
Randomly distributed	Round/Hetero	0.95
Equally distributed	Random/Round	**10.11**
Equally distributed	Random/Hetero	**3.15**
Equally distributed	Round/Hetero	**6.35**

4.3 Summary of Results

To sum up the results, there are several points that can be noted:

1. The starting locations of agents in a network formation simulation is a strong factor. I chose four types of starting location options and found the center and the corner starting locations are alike and the randomly distributed and equally distributed starting locations are alike. But these two sets of starting locations also have significant differences on some network measures. This suggested that the choice of starting locations of agents is important and needs to be considered and reported when designing experiments.

2. Different types of agent models form different networks, and heterogeneous agents with each agents at 50% do not give an average value of network properties. This shows that heterogenous agents change the network formation fundamentally compared to sets of homogeneous agents, and are not simply averages of the homogeneous network values.

The next chapter is the summary of this thesis, and it explains the limitations of the current work, and provides some possible future work.

Chapter 5

Conclusion and Discussion

In this chapter, I give a summary of the experiments, and explain the limitation of the experiments and the future work that I look forward to be done.

5.1 Summary

In this thesis, I set up an environment to run cognitive-agent-based network formation simulation, made two different models, ran experiments with both homogeneous agent sets and heterogeneous agent sets in different starting locations, and analyzed the results.

Experiment 1. Experiment 1 shows that the starting locations of agents influence the network formation strongly. Four kinds of starting locations were tested. I found the data sets with the center starting location and the corner starting location have similar results and the randomly distributed starting locations and the equally distributed starting locations have similar results.

For the edge count, the mean value of the randomly distributed starting locations and the equally distributed starting locations are around 200 edges, where the mean value of the center starting location and the corner starting location are around 600 edges.

For assortativity, the results of the randomly distributed starting locations and the equally distributed starting locations have assortativity, but the results of the center starting location and the corner starting location have disassortativity.

For global cluster coefficient, the mean value of the randomly distributed start-

ing locations and the equally distributed starting locations are around 0.7, where the mean value of the center starting location and the corner starting location are around 1.

For diameter, the mean value of the randomly distributed starting locations and the equally distributed starting locations are around 5, where the mean value of the center starting location and the corner starting location are around 12.

Reciprocity is the only value that the results of the corner starting location is different from other three, where the mean value is 0.65, and the other three are between 0.45 and 0.5.

With the results, it shows that when doing a network formation simulation with spatial settings, it is required to consider and report the starting locations of agents and how they are distributed.

Dunbar [1998] claims that on average we have 150 friends memorized. If we take the number into consideration and want a normalized simulation, then the results of putting agents distributed on a map is closer to it. On the other hand, putting agents together simulates another scenario.

Experiment 2. Experiment 2 shows that heterogeneous agents in comparison with homogeneous agents change the network formation, though the effect on edges count, assortativity, global cluster coefficient, diameter, and reciprocity is not as strong as changing starting locations.

If I compare the property values between agent model sets, I found that putting two types of agents together with half and half percentage, the properties of the network graph are not simply the average of networks with two types of homogeneous agents. Moreover, the values from heterogeneous agents do not always fell between two sets of homogeneous agents.

For the edge count, the mean value of the randomly distributed starting locations and the equally distributed starting locations are around 200. With the center starting location and the corner starting location, the values have variances between different agent sets. Heterogeneous agents formed fewer edges than the other two types of agents alone.

For assortativity, most of the results of the randomly distributed starting locations and the equally distributed starting locations have assortativity, but the results of the center starting location and the corner starting location have disas-

sortativity. Round-walk agents with the corner starting location is an exception, where the value possible to be associative and dissociative. But homogeneous agents make networks less associative for both associativity and disassociativity.

For global cluster coefficient, the mean value of the randomly distributed starting locations and the equally distributed starting locations are between 0.5 and 0.7, where the mean value of the center starting location and the corner starting location are around 1. The heterogeneous agents lower the value of global cluster coefficient except when with the equally distributed starting locations.

For diameter, with the center starting location, heterogeneous agents have value between homogeneous agents; With the corner starting location, heterogeneous agents have value bigger than homogeneous agents; With randomly distributed starting locations, heterogeneous agents have value smaller than homogeneous agents; With equally distributed starting location, heterogeneous agents have values between homogeneous agents but closer to the random-walk agents.

For reciprocity, with the center starting location, heterogeneous agents have value between homogeneous agents but closer to the round-walk agents; With the corner starting location, heterogeneous agents have a smaller value than homogeneous agents; With randomly distributed starting locations, heterogeneous agents have a smaller value than homogeneous agents; With equally distributed starting location, heterogeneous agents have a value between homogeneous agents but closer to the random-walk agents.

This suggests heterogeneous agents not only change the result of network formation, and the introduction of heterogeneous agents gave me a different network formation. Using homogeneous agents appears to not represent diversity in agents when doing simulation and variances of agents needs to be considered. This also suggests that if we run the simulation with the agent model set that do not reflect the real condition, the simulation may fail to represent the real cases, and the realism of modeling is always a target to pursue.

So, this work shows that heterogeneity of agents is important, it influences edges, assortativity, global cluster coefficient, diameter, and reciprocity of a network. Heterogeneous agents should be used to model humans if they are diverse.

5.2 Limitations and Future Work

Below are some limitations of this work and some future improvements.

Modularization. The current models of ACT-R have duplicate code if I made a new model, especially the productions and the chunk types setting related to ACT-R. This is because ACT-R is a low-level language compared to other programming languages [ACT-R Research Group, 2012]. According to software development principles, it is a good practice to avoid duplicate code. To make the code reusable, finding a way to put productions in the models into modules is a future direction. Cohen et al. [2005] introduced a high-level language, Herbal, that can also compile into Jess [Friedman-Hill, 2002] or Soar [Lehman et al., 2007]. It let developers focus onknowledge representation rather than details of cognition modeling. If Herbal can support ACT-R and TAWA better, it is a good direction to remodel the current models.

Scaling. In this research, the experiments are running with two servers: one for the VIPER server and one for agents. If the environment is moved to a cloud or HPC (High-Performance Computing), it is possible to scale up experiments with more agents at the same time. Right now, there is a hard boundary of memory that an affordable server can provided. With HPC, I can simulate a larger scale of agents and examine larger network formations.

But in such case, I may have to solve some new problems introduce with the new infrastructure, such as the network latency between servers and the synchronization problem in a distributed system. However, having different numbers of agents is expected to form a different network, and with the increased amount of agents, I can expect more realistic results.

VIPER. VIPER is a flexible system that is possible to adopt different scenarios and form different kind of experiments. There are still many possible social network formation simulations that can be done with VIPER, such as:

1. Use more realistic maps that reflect the real world and make the agents perform more realistic behaviors with better timing assumptions. The current setting of the map is a well connected 5 x 5 grid that can be regarded as a perfect environment with boundaries. For a real world case, we have routes connect between places and people tend to stick around some locations. With

this settings, we may not only simulation network formations but predict the locations that people tend to gather together and form a town. An example of a more realistic behavior is that an agent will spend more time on a friend if it knows this friend better. This can be done by the activation value of the chunk to memorize this friend and adding the timing constraint.

2. It is also possible to apply to the Predator/Prey domain, which is used in Stone and Veloso [2000]. In the Predator/Prey domain, or "Pursuit" domain, there are usually four predators and one prey. The goal of predators is to capture the prey. Predators can see and talk to each other, where the prey move randomly and sometimes stay at the same point. Because there are several possibilities to apply agent models to the predators, it is a domain that can be used to study agents and compare the results between homogeneous and heterogeneous agents or agents with different strategies. Also, what will happen if there are multiple groups of Predator/Prey on the same map?

3. Make the agents have the ability to communicate with each other and exchange information. There are numerous possibilities of the information, such as the locations where an agents met a lot of other agents, or expressing personality. Then it is possible to do simulation with homophily [McPherson et al., 2001]. The agent models I use now do not differentiate between themselves except by names, and they do so completely uniquely. So it is possible to add characteristics, such as using colors to decorate agents, and have cognitive productions that urge agents to find other agents in the same color. Some work has been done in this topic, such as Schelling [1971], Möbius and Rosenblat [2001].

4. Have goals for each agent. There are several possibilities of goals, for example, all agents have the same goal to cooperate, all agents have their own private goals, or two groups of agents have different goals. The last one can be regarded as another kind of heterogeneity. This can improve our knowledge with team work and groups, which is suggested in [Ritter and Norling, 2006; Zacharias et al., 2008].

References

ACT-R Research Group (2012). ACT-R 6 tutorial. `http://act-r.psy.cmu.edu/actr6/units.zip`.

Albert, R. and Barabási, A.-L. (2002). Statistical mechanics of complex networks. *Reviews of Modern Physics*, 74(1):47.

Anderson, J. R. (2007). *How can the human mind occur in the physical universe?* Oxford University Press, New York, NY.

Anderson, J. R., Bothell, D., Byrne, M. D., Douglass, S., Lebiere, C., and Qin, Y. (2004). An integrated theory of the mind. *Psychological Review*, 111(4):1036–1060.

Carley, K. and Newell, A. (1994). The nature of the social agent. *Journal of Mathematical Sociology*, 19(4):221–262.

Carley, K. M. and Gasser, L. (1999). Computational organization theory. *Multiagent Systems: A modern approach to distributed artificial intelligence*, pages 299–330.

Cohen, M. A., Ritter, F. E., and Haynes, S. R. (2005). Herbal: A high-level language and development environment for developing cognitive models in soar. In *Proceedings of the 14th Conference on Behavior Representation in Modeling and Simulation*, pages 177–182. 05-BRIMS-044. Orlando, FL: U. of Central Florida.

Davidsson, P. (2001). Multi agent based simulation: Beyond social simulation. In *Multi-Agent-Based Simulation: Second International Workshop, MABS 2000, Boston, MA, USA, July 2000; Revised and Additional Papers*, volume 1979, pages 97–107, New York, NY. Springer.

Dunbar, R. I. M. (1998). *Grooming, gossip, and the evolution of language*. Harvard University Press, Cambridge, MA.

Easley, D. and Kleinberg, J. (2010). *Networks, Crowds, and Markets*. Cambridge University Press, New York, NY.

Fan, X. and Yen, J. (2004). Modeling and simulating human teamwork behaviors using intelligent agents. *Physics of Life Reviews*, 1(3):173–201.

Friedman-Hill, E. (2002). Rule engine API specification (JSR 94). `http://jcp.org/aboutJava/communityprocess/review/jsr094/`.

Garlaschelli, D. and Loffredo, M. I. (2004). Patterns of link reciprocity in directed networks. *Physical Review Letters*, 93(26):id.268701.

Hiam, J., Zhao, C., and Ritter, F. E. (2011). Viper: A text based environment for intelligent agents. Technical report, `http://acs.ist.psu.edu/reports/hiamZR11.pdf`.

Hollis, G. (2009). NakedMud. `http://homepages.uc.edu/~hollisgf/nakedmud.html`.

Kaulakis, R., Zhao, C., Morgan, J. H. H., W, J., Sanford, J. P., and Ritter, F. E. (2012). Defining factors of interest for large-scale socio-cognitive simulations. In *Proceedings of ICCM - 2012- Eleventh International Conference on Cognitive Modeling*, pages 117–122, Berlin. Universitaetsverlag der TU Berlin.

Kossinets, G. and Watts, D. J. (2006). Empirical analysis of an evolving social network. *Science*, 311(5757):88–90.

Laird, J. E. (2012). *The Soar Cognitive Architecture*. MIT Press, Cambridge, MA.

Lehman, J., Laird, J., and Rosenbloom, P. (2007). A gentle introduction to soar, an architcture for human cognition: 2006 update. `http://ai.eecs.umich.edu/soar/sitemaker/docs/misc/GentleIntroduction-2006.pdf`.

Macal, C. M. and North, M. J. (2005). Tutorial on agent-based modeling and simulation. In *Proceedings of the 37th conference on Winter simulation*, pages 2–15, Washington, D.C. Winter Simulation Conference.

Mannix, E. and Neale, M. A. (2005). What differences make a difference? the promise and reality of diverse teams in organizations. *Psychological Science in the Public Interest*, 6(2):31–55.

McPherson, M., Smith-Lovin, L., and Cook, J. M. (2001). Birds of a feather: Homophily in social networks. *Annual Review of Sociology*, 27:415–444.

Möbius, M. M. and Rosenblat, T. S. (2001). The process of ghetto formation: Evidence from chicago. Working paper. `http://trosenblat.surveyor.nber.org/papers/Files/Chicago/chicago-dec7.pdf`.

Morgan, J. H., Morgan, G. P., and Ritter, F. E. (2010). A preliminary model of participation for small groups. *Computational & Mathematical Organization Theory*, 16(3):246–270.

Newell, A. (1973). You can't play 20 questions with nature and win: Projective comments on the papers of this symposium. Technical report, Computer Science Department. Paper 2033. `http://repository.cmu.edu/compsci/2033/`.

Newman, M. E. (2002). Assortative mixing in networks. *Physical Review Letters*, 89(20):id. 208701.

Newman, M. E. (2003). The structure and function of complex networks. *SIAM Review*, 45(2):167–256.

Ritter, F. E. and Norling, E. (2006). Including human variability in a cognitive architecture to improve team simulation. pages 417–427, Cambridge, UK. Cambridge University Press.

Ritter, F. E., Reifers, A., Klein, L. C., Quigley, K., and Schoelles, M. (2004). Using cognitive modeling to study behavior moderators: Pre-task appraisal and anxiety. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, volume 48, pages 2121–2125, Santa Monica, CA. SAGE Publications.

Schelling, T. C. (1971). Dynamic models of segregation. *Journal of Mathematical Sociology*, 1(2):143–186.

Shoham, Y. (1993). Agent-oriented programming. *Artificial Intelligence*, 60(1):51–92.

Stone, P. and Veloso, M. (2000). Multiagent systems: A survey from a machine learning perspective. *Autonomous Robots*, 8(3):345–383.

Sun, R. (2009). Cognitive architectures and multi-agent social simulation. In *Multi-Agent Systems for Society*, pages 7–21. Springer.

Sycara, K. P. (1998). Multiagent systems. *AI Magazine*, 19(2):79–92.

Wasserman, S. and Faust, K. (1994). *Social network analysis: Methods and applications*, volume 8. Cambridge University Press, New York, NY.

Watts, D. J. (1999). *Small worlds: The dynamics of networks between order and randomness*. Princeton University Press, Princeton, NJ.

Wooldridge, M., Jennings, N. R., et al. (1995). Intelligent agents: Theory and practice. *Knowledge Engineering Review*, 10(2):115–152.

Zacharias, G. L., MacMillan, J., Van Hemel, S. B., et al. (2008). *Behavioral modeling and simulation: From individuals to societies*. National Academies Press, Washington, D.C.

Zachary, W. W. (1977). An information flow model for conflict and fission in small groups. *Journal of Anthropological Research*, 33(4):452–473.

Zhao, C., Hiam, J. W., Morgan, J. H., and Ritter, F. E. (2011). A multi-strategy spatial navigation model in a text-based environment. In *Proceedings of the 20th Conference on Behavior Representation in Modeling and Simulation*, pages 251–258. 11-BRIMS-036.

Zhao, C., Kaulakis, R., Morgan, J. H., Hiam, J., Ritter, F. E., Sanford, J., and Morgan, G. (2013). Building social networks out of cognitive blocks: Factors of interest in agent-based socio-cognitive simulations (in press). *Computational and Mathematical Organization Theory*.

Zhao, C., Kaulakis, R., Morgan, J. H., Hiam, J. W., and Ritter, F. E. (2012a). Modeling a cognitively limited network in an agent-based simulation. In *Proceedings of the 34th Annual Conference of the Cognitive Science Society*, pages 2603–2608, Austin, TX. Cognitive Science Society.

Zhao, C., Kaulakis, R., Morgan, J. H., Hiam, J. W., Sanford, J. P., Ritter, F. E., and Morgan, G. P. (2012b). Socio-cognitive networks: Modeling the effects of space and memory on generative social structures. In *Proceedings of the 21st Conference on Behavior Representation in Modeling and Simulation*, pages 24–31, Amelia Island, FL. BRIMS Society.