

Simulating the Formation of Transactive Memory Systems

X. Yuan

George Mason University

A. N. Devereaux

George Mason University

Abstract

A transactive memory system (TMS) is a system where members share the knowledge of “who knows what.” A large volume of studies has been done on how social networks relate to the formation and dynamics of TMSs in workplaces. However, previous research tends to focus on dyadic network ties instead of triadic network structures. This paper extends research on how certain triadic microstructures relate to the formation of TMSs using a model of task completion within a team. By comparing the networks produced by our model with random networks, we are able to determine that our networks are similar to real-life TMS networks as examined in the literature. Furthermore, we show that larger TMS networks start to resemble random networks of the same size, which may provide further intuition for why too-large teams tend to suffer losses in productivity.

Keywords

transactive memory, social network analysis, triadic structures, team performance

1. INTRODUCTION

A transactive memory system (TMS) is a system in which members of a team construct a cognitive division of labor based on shared awareness of “who knows what” (Peltokorpi, 2008). There are two essential components for a cognitive division of labor: 1) internal memory: knowledge that the individual stores in their own brain, and 2) external memory: information about knowledge that is stored in other people’s brains. Transactive memory is a shared, collective kind of knowledge. When transactive memory is needed to complete tasks, members in a transactive memory system know who can help them complete their tasks. Organizational team members vary in their ability to accomplish tasks in the workplace. The efficient and accurate recognition, distribution,

and transfer of knowledge within teams indicates better team performance. In any work based on knowledge, expertise is proven to be one of the most important resources (Anand, Manz & Click, 1998; Argote & Ingram, 2000; Faraj & Sproull, 2000). A TMS is the knowledge stored in each team member's memory combined with information of different teammates' domains of expertise (Wegner, 1995).

The formation and dynamics of TMS involves three simultaneous processes:

- 1) directory updating and expertise recognition. Team members use expertise directory to identify knowledge resources in the team for task completion;
- 2) communication to allocate information. Team members use expertise directories to allocate information to the person who is the team's domain expert; and
- 3) communication to retrieve information.

Current research places significant emphasis on the process of TMS creation, since in this process, the benefits of a TMS become evident to team members (Wegner et al., 1985; Palazzolo, 2005; Palazzolo, et al., 2006). A TMS is by nature a social network, and by studying a TMS as a network, we can better determine how TMSs form (Monge & Contractor, 2003; Rulke & Galaskiewicz, 2000).

Our project is about creating a model that accurately simulates the formation of a TMS. In order to determine whether or not the networks produced by our model are TMS-like, we need to determine whether the network microstructures correlated with real-world TMSs are present in the networks produced by our model.

2. LITERATURE REVIEW

Transactive memory was first introduced in the context of intimate couples. Wegner (1986) explored how intimate couples form a cognitive division of labor to solve problems. The concept has been applied in other contexts, especially within organizational studies that focus on how transactive memory systems (TMS) lead to better group performance in workplaces. Social network analysis has been foundational in the study of TMS. The majority of these studies, however, examine TMS from the perspective of dyadic network relations (e.g., strength of ties) instead of triadic network structures (e.g., transitive triads). Research on network triads and TMS such as Palazzolo (2005) and Lee, et al. (2014) lay important theoretical groundwork for our simulation model.

2.1 Transactive Memory System and Social Network Analysis

Group members' network ties play an important role in expert identification and information allocation and retrieval. Network ties within the group inform group members who they can reach out to and how they can work with other team members efficiently. Borgatti and Cross (2003) examine how network ties relate to information seeking. The process of TMS development, especially the stage of building expert directory, depends on the relational characteristics. They find three relational characteristics are predictive of the behavior of information seeking: 1) how much does the individual know about the expertise of others; 2) how does the individual perceive the expertise of others; 3) how accessible the experts are that the individual wants to reach out to. Similarly, Yuan et al. (2010) demonstrated that higher team communication tie

strength happens more in well-developed TMS expertise recognition. Moreover, there's a positive correlation between well-developed TMSs and better team performances.

Network centralities are also relevant to accurate group expertise recognition. Su (2012) found that a group member's accuracy in expertise recognition was positively influenced by one's degree centrality in the communication network. However, if someone is doing work remotely, such an influence doesn't exist.

Triadic micro-level network structures, not just dyadic ties, are relevant to TMS development. Palazzolo (2005) found that better information retrieval in work teams is correlated with high likelihood of in-star, out-star, and transitive triad microstructures compared to random networks of the same size. Lee et al. (2014) extended Palazzolo's research and demonstrated that transitive triads are associated with mature TMSs and network density has dual effects on the development of TMSs because more communication is beneficial to TMS development only under the presence of transitive triads.

Although Lee et al (2014) examined the positive correlation between TMSs and transitive triads through rich empirical data, no agent based simulations exist that have demonstrated how microstructures associated with TMSs arise in the process of individual agents completing tasks by coordinating with other agents who also need to complete tasks. The purpose of our investigation was to provide a simulation environment in which TMSs emerge from the necessity of agents to coordinate with others in order to complete tasks.

2.2 Examining Transactive Memory Systems through Computational Models

As a large amount of research has been done through collecting empirical data to explore the relationship between TMS and network ties and structures, Palazzolo (2006) proposed that through computational models, we can examine TMS under different circumstances and start points to understand TMS in a new way. By controlling different conditions and scenarios in computational models, Palazzolo found that initial knowledge, initial accuracy of expertise recognition, and network size has impact on communication density, and subsequently impact the development of TMS.

However, Palazzolo only focuses on the relationship between network level statistics and TMS development instead of micro-level triadic structures. Moreover, as mentioned, dyadic network structure and TMS is only examined through empirical data rather than computational methods. This paper aims to combine these two, that is to apply the computational model method to examine how triadic structure relates to the development of TMS in different circumstances.

2.3 Motivational Framework: Agent-based Computational Economics

Agent-based computational economics (ACE) is a methodological framework that seeks to answer questions with positive economic content using agent-based modeling (LeBaron & Tesfatsion, 2008; Axtell, 2000, 2005). As such, our motivation for simulating transactive memory systems is to better understand the innerworkings of systems that divide knowledge, or what some call the cognitive division of labor.

The ACE approach to understanding the cognitive division of labor can be contrasted to traditional optimization approaches whereby individuals who wish to

contribute knowledge to a project calculate an expected marginal benefit and marginal cost of their contribution; if the marginal benefit of their contribution is expected to exceed the marginal cost, they contribute (Kitcher, 1990, 1993; Strevens, 2003). Other agent-based models of the cognitive division of labor have deviated in their own way from the traditional optimization framework, for instance, to view the cognitive division of labor like a strategic search over an epistemic landscape (Weisberg & Muldoon, 2009).

Generally, it behooves us in our understanding of the cognitive division of knowledge to go beyond classical representations of Robinson Crusoe's search for knowledge in autarky, to the problem-solver embedded in a social substructure. Agents who divide their knowledge with each other do so in the context of an existing social network, in a way that results in the creation of a new and perhaps special kind of social network. It is the latter point we wish to make with our simulation of the creation of transactive memory systems: that the individual act of attempting to complete a task entails making particular network connections with experts and coordinators. That is, the topology of transactive memory systems as observed empirically can be modeled as a result of the desire to divide cognitive labor. Thus, we provide a bottom-up explanation for the type of teams that organize to cognitively divide their labor with each other.

3. MODEL

The model was written to simulate the formation of transactive memory systems (TMSs) in NetLogo. The flow of the model is similar conceptually to Hauke (2014). A certain small number of nodes are initialized without edges. The node properties are *expertise*, a list of numbers normally distributed from 0 to 1 that is the same length as the

number of areas in which agents can have expertise. For instance, if there are four areas of expertise, agent i 's $expertise_i$ might look like:

$$expertise_i = [0.2345, 0, 1, 0.071]$$

More generally, for m areas of expertise, the expertise of agent i is:

$$expertise_i = [a_0^i, a_1^i, a_2^i, \dots, a_{m-1}^i]$$

where $expertise_i$ is a vector of length m , and $a_j^i \sim N(0,1)$ and bounded between 0 and 1.

The model is initialized with a set of tasks, of a user-determined length, that each agent endeavors to complete. The same set of tasks is given to every agent to complete: *tasks-to-complete* is initialized as a global variable of the model. Given m areas, *tasks-to-complete* is a list of random integers between 0 and $m-1$ of user-given length. The random choice of an integer between 0 and $m-1$ is executed at each step of building the list. An example *tasks-to-complete* list of length 10, given 4 areas of expertise where the integers $\{0, 1, \dots, 9\}$ refer to which area of expertise is required to complete the task, might look something like this:

$$tasks-to-complete = [3, 0, 2, 2, 1, 0, 0, 3, 1, 1]$$

Agents must complete *tasks-to-complete* in order. When agents are activated, they attempt to complete the first task in their personal copy of *tasks-to-complete*. After completing a task, they remove the completed task from the list.

Agents can complete a task by one of two methods. First, if they have an expertise of 1 in the given task area, they can complete a task themselves with no help from other agents in the space/network. For instance, in our example above, agent i has an expertise of 1 in expertise area 2, and so would be able to complete any task 2's in her *tasks-to-complete* list without assistance. Second, if an agent is unable to complete her

task due to insufficient expertise, she may either ask an existing area-expert in her network to complete the task, or she may seek assistance by asking agents with which she doesn't yet have a network connection whether or not they can help her complete the task.

To complete a task with the help of another agent, the sum of the ego agent's area-expertise and the alter agent's area-expertise must be greater than or equal to 1. That is, say agent i 's *expertise* for m areas is:

$$expertise_i = [a_0^i, a_1^i, a_2^i, \dots, a_{m-1}^i]$$

and agent j 's *expertise* for m areas is:

$$expertise_j = [a_0^j, a_1^j, a_2^j, \dots, a_{m-1}^j]$$

Then in order for agent i to complete task k with agent j 's help:

$$a_k^i + a_k^j \geq 1$$

If agent j is able to help agent i complete her task, a link is forged between the two agents, and the completed task is dropped from the agent's personal copy of *tasks-to-complete*. If agent i needs to complete another task k further down in her *tasks-to-complete*, she first searches her existing network to see if an agent with expertise in k exists. If he does, then the completed task is dropped from her list.

We can think of the agent's existing network as her **directory** of experts. Whenever she adds a new expert, she is **updating her directory**. Since directory updating is a key feature of the formation and maintenance of transactive memory systems (Palazzolo, etc), this behavior is built directly into the model itself.

3.1 NetLogo: description of graphics, layout, and monitors

Agent nodes are initialized as blue disks. Edge colors in the model are delineated by area of expertise. Area 0 may be associated with red, area 1 with yellow, and so on. Whenever an agent makes a successful connection with another agent, the area of expertise that agent helped the ego agent complete is colored to match that area of expertise. Once an agent has completed her copy of *tasks-to-complete*, her node color is updated to “white” in the model.

The model is available to download from abigaildevereaux.com.

3.2 Parameters of the model

There are three model parameters adjustable by the user:

- How many nodes are initialized in that run of the model (*num-nodes*)
- How many areas of expertise are present in that run of the model (*num-areas*)
- How long *tasks-to-complete* should be (*task-length*)

In Figure 1, we show the model’s control panel.

The monitor in the bottom left of the control panel tracks how many individual tasks were completed by all the agents in the network. That is, each time an agent completes a task from their personal copy of *tasks-to-complete*, the monitor value “tasks completed” increments by one.

While the visualizations of the network are interesting in the model itself, they are not crucial or even sufficient for determining whether transactive memory systems are being formed by the model. In order to determine whether the model does in fact result in the formation of TMS-like structures that divide cognitive labor in the way specified by (cite, cite, cite), we need to test the networks outputted by the model against a few hypotheses.

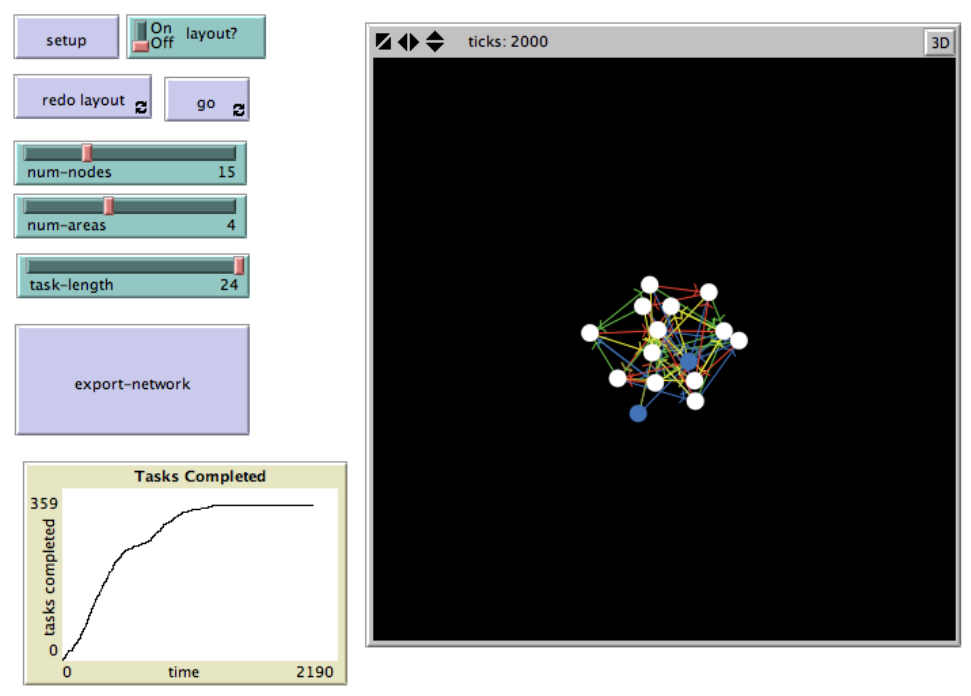
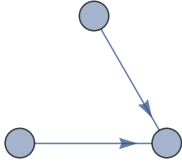


Figure 1: “Simulating Transactive Memory Systems,” NetLogo.

4. HYPOTHESES

Our main hypothesis is that we will see more of the microstructures associated with TMSs in the literature, and fewer of the microstructures negatively associated with TMSs. The network microstructures of interest are listed, with the direction of their correlation, in Table 1 below.

Microstructure	Visualization	Correlation	Supporting Literature
In-star: Two agents connected to the same agent but not to each other		Positive	Palazzolo, 2005

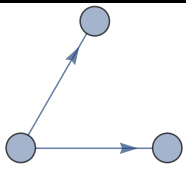
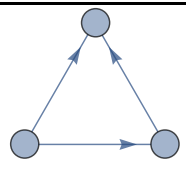
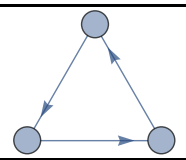
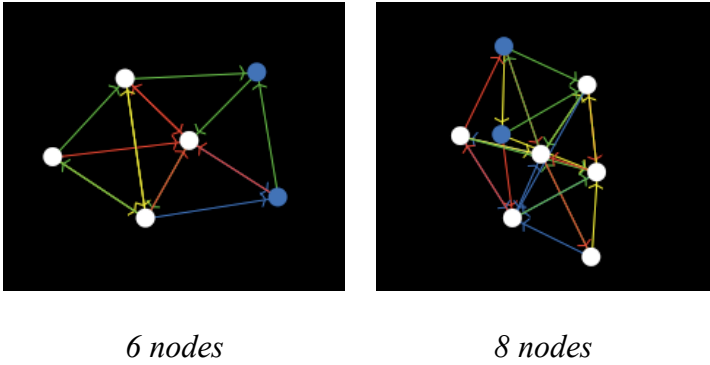
Out-star: One agent connected to two different agents who aren't connected to each other		Positive	Palazzolo, 2005
Transitive triad: One agent (the tertius) is connected to two agents who are also connected		Positive	Palazzolo, 2005; Lee, Bachrach, & Lewis, 2014
Cyclic triad: Each agent is connected to only one other agent in the triad		Negative	Lee, Bachrach, & Lewis, 2014

Table 1: Microstructures correlated with transactive memory systems.

5. RESULTS

5.1 Networks

We studied 6, 8, 12, and 30-node networks. Figure 2 is a table of the networks, with a typical representation of how they looked, by the number of nodes in the network.



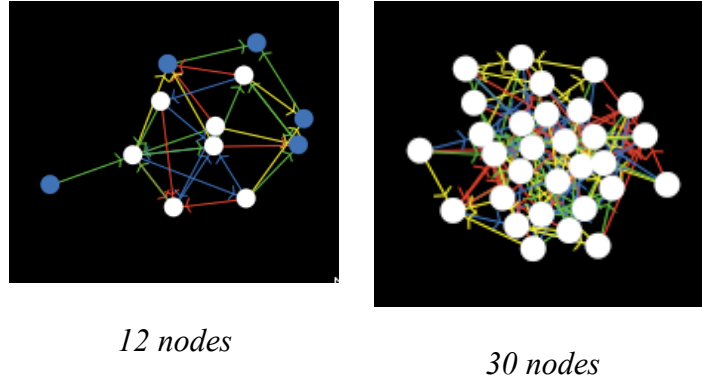


Figure 2: Typical TMS networks for 6, 8, 12 and 30 nodes

5.2 Triadic census

We conducted a triadic census on our TMS graphs using the NetworkX package in Python, and on random directed graphs of the same size, and compared which graphs have a greater prevalence of structures associated with TMSs. We ran the model for 1000 runs per node number, with the default of 4 task areas and a 24-long set of tasks for each node to complete.

The triadic census algorithm being used is described in Batagelj and Mrvar (2001). Figure 3 shows a list of triads, with associated codes, being searched for by the algorithm.

Triads associated with TMSs are transitive triads, 030T above, in-stars, or 021U above, and out-stars, or 021D above (Palazzolo, 2005; Lee, Bachrach, and Lewis, 2014). 030C, triadic cycles, are negatively correlated with TMSs (Lee, Bachrach, and Lewis, 2014).

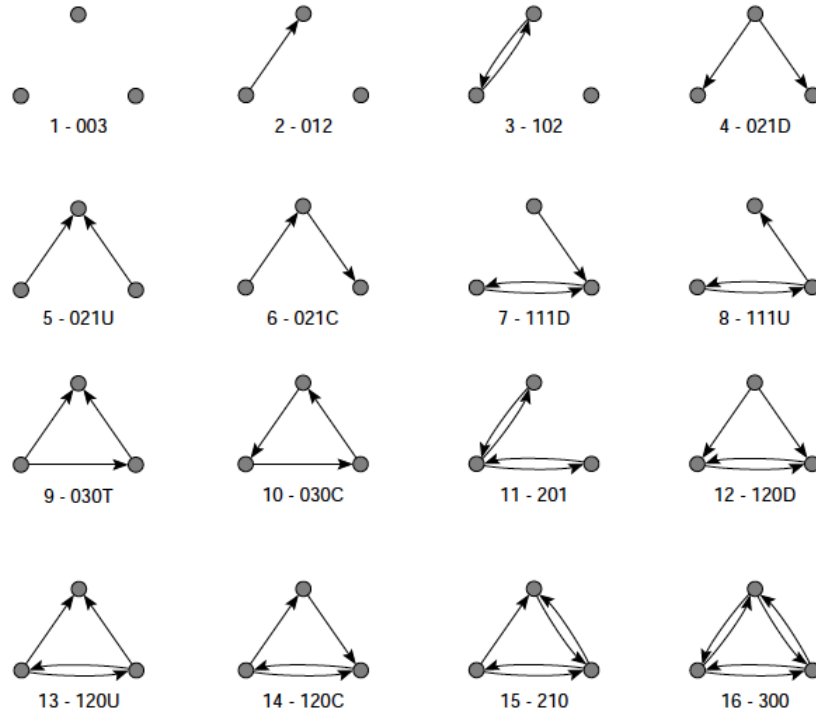


Figure 3: Types of triads. Figure taken from Batagelj and Mrvar (2001).

5.3 Regression

Our regression models look for linear correlations between TMSs and the triadic microstructures of interest. In order to discover the correlation between these triadic microstructures and TMS networks, we develop a dummy variable called *TMS*, where $TMS = 1$ if the network in question was outputted from the simulation developed above, and $TMS = 0$ if the network is a random network with the same number of nodes and edges (if specified).

We conducted 1000 runs per node number (= 6, 8, 12, 30) in NetLogo's BehaviorSpace.

The data used in the regression is built in Python; the adjacency matrices of the

TMS networks are imported into Python, in which we can conduct a triadic census on each of them. We determine and store the edges counts of the TMS networks imported into Python, then build random networks with the same nodes and edge numbers as the imported TMS networks, so that we're comparing oranges to oranges in the regression analysis. We rely mostly on built-in functions of the NetworkX package and only minimally process the networks by, for instance, leaving out disconnected networks from the analysis. After the data is built, it is exported to STATA for regression.

The regression models under consideration must be carefully constructed so that there are no overlapping structures, since then we would be running into the problem of collinearity between the independent variables. In order to avoid collinearity, we look at several regression models in order to isolate effects related to the microstructures of interest in our hypotheses.

Regression 1: Transitive triads and triadic cycles

The first regression model looks at the relationship between TMSs (the dependent variable), and transitive triads and triadic cycles (the independent variables).

$$TMS_i = \beta_0 + \beta_1 (\text{transitive triad})_i + \beta_2 (\text{triadic cycle})_i + \varepsilon_i \quad (\text{REG1})$$

where $(\text{transitive triad})_i$ = the number of 030T structures in network i , and $(\text{triadic cycle})_i$ = the number of 030C structures in network i , and ε_i is a mean-zero error term.

In all the tables below, we regress on edge subsets, shown as the first columns, and over all edges, shown as the last column.

Results for 6 nodes:

triadic microstructure	13 edges	12 edges	11 edges	10 edges	all edges
transitive triads (030T)	0.0860***	0.107***	0.132***	0.155***	0.0700***
	(44.62)	(43.74)	(34.92)	(25.31)	(35.03)
triadic cycles (030C)	-0.0439**	-0.104***	-0.150***	-0.180***	-0.237***
	(-3.01)	(-7.87)	(-7.51)	(-7.58)	(-16.80)
_cons	-0.127***	-0.0946***	-0.0798***	-0.0290	0.262***
	(-6.71)	(-4.97)	(-3.44)	(-0.97)	(20.56)
<i>N</i>	188	274	314	236	1810

t statistics in parentheses

* $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$

Table 2: Regression table output for 6 nodes. Edge subsets are shown in the first columns, and the result taken over all edges is shown in the last column.

Results for 8 nodes:

triadic microstructure	20 edges	19 edges	18 edges	17 edges	all edges
transitive triads (030T)	0.0559***	0.0638***	0.0679***	0.0799***	0.0452***
	(49.73)	(41.44)	(29.93)	(31.21)	(45.08)
triadic cycles (030C)	-0.0532***	-0.0759***	-0.0929***	-0.0990***	-0.162***
	(-6.04)	(-9.14)	(-7.12)	(-6.49)	(-23.09)
_cons	-0.134***	-0.112***	-0.0681*	-0.0582*	0.209***
	(-7.26)	(-5.21)	(-2.40)	(-2.21)	(17.17)
<i>N</i>	252	286	224	234	1956

t statistics in parentheses

* $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$

Table 3: Regression table output for 8 nodes. Edge subsets are shown in the first

columns, and the result taken over all edges is shown in the last column.

Results for 12 nodes:

triadic microstructure	35 edges	33 edges	31 edges	29 edges	all edges
transitive triads (030T)	0.0347***	0.0371***	0.0447***	0.0499***	0.0268***
	(30.75)	(28.25)	(18.48)	(16.40)	(46.79)
triadic cycles (030C)	-0.0569***	-0.0691***	-0.0585***	-0.0951***	-0.117***
	(-9.89)	(-10.41)	(-5.11)	(-7.55)	(-34.41)
_cons	-0.115***	-0.0469	-0.0995*	-0.0123	0.224***
	(-3.61)	(-1.45)	(-2.00)	(-0.24)	(17.39)
N	214	232	128	100	1994

t statistics in parentheses

* $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$

Table 4: Regression table output for 12 nodes. Edge subsets are shown in the first columns, and the result taken over all edges is shown in the last column.

Results for 30 nodes:

triadic microstructure	105 edges	102 edges	99 edges	96 edges	all edges
transitive triads (030T)	0.0192***	0.0164***	0.0186***	0.0230***	0.0154***
	(11.17)	(11.70)	(11.82)	(8.84)	(35.76)
triadic cycles (030C)	-0.0407***	-0.0631***	-0.0632***	-0.0702***	-0.0680***
	(-8.35)	(-15.98)	(-17.03)	(-10.48)	(-56.25)
_cons	-0.0993	0.131	0.135*	0.0266	0.237***
	(-1.07)	(1.92)	(2.03)	(0.27)	(12.62)
N	64	194	182	114	2000

t statistics in parentheses

* $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$

Table 5: Regression table output for 30 nodes. Edge subsets are shown in the first columns, and the result taken over all edges is shown in the last column.

Regression 2: In-stars and out-stars

The second regression model looks at the relationship between TMSs (the dependent variable), and in-stars and out-stars (the independent variables).

$$TMS_i = \beta_0 + \beta_1 (in-star)_i + \beta_2 (out-star)_i + \varepsilon_i \quad (\text{REG2})$$

where $(in-star)_i$ = the number of 021U structures in network i , and $(out-star)_i$ = the number of 021D structures in network i , and ε_i is a mean-zero error term.

Results for 6 nodes:

triadic microstructure	13 edges	12 edges	11 edges	10 edges	all edges
in-stars (021U)	0.125***	0.126***	0.106***	0.124***	0.0998***
	(5.83)	(8.84)	(8.15)	(8.66)	(16.42)
out-stars (021D)	0.0993***	0.129***	0.117***	0.105***	0.109***
	(4.55)	(8.61)	(9.62)	(6.86)	(18.41)
_cons	0.125	-0.0342	0.00643	0.0154	0.101***
	(1.96)	(-0.67)	(0.14)	(0.31)	(5.22)
N	188	274	314	236	1810

t statistics in parentheses

* $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$

Table 6: Regression table output for 6 nodes. Edge subsets are shown in the first columns, and the result taken over all edges is shown in the last column.

Results for 8 nodes:

triadic microstructure	20 edges	19 edges	18 edges	17 edges	all edges
in-stars (021U)	0.0686***	0.0704***	0.0647***	0.0586***	0.0615***
	(14.65)	(13.95)	(10.87)	(8.71)	(26.34)
out-stars (021D)	0.0599***	0.0630***	0.0672***	0.0724***	0.0609***
	(11.67)	(11.94)	(11.15)	(11.27)	(25.67)
_cons	-0.309***	-0.334***	-0.312***	-0.261***	-0.178***
	(-7.04)	(-7.62)	(-6.08)	(-5.03)	(-9.78)
N	252	286	224	234	1956

t statistics in parentheses

* $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$

Table 7: Regression table output for 8 nodes. Edge subsets are shown in the first columns, and the result taken over all edges is shown in the last column.

Results for 12 nodes:

triadic microstructure	35 edges	33 edges	31 edges	29 edges	all edges
in-stars (021U)	0.0260***	0.0297***	0.0256***	0.0279***	0.0238***
	(16.42)	(17.41)	(9.31)	(9.37)	(28.56)
out-stars (021D)	0.0269***	0.0250***	0.0349***	0.0331***	0.0225***
	(17.63)	(16.17)	(12.90)	(9.22)	(26.93)
_cons	-0.661***	-0.646***	-0.671***	-0.630***	-0.380***
	(-17.70)	(-15.91)	(-10.62)	(-9.65)	(-22.28)
N	214	232	128	100	1994

t statistics in parentheses

* $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$

Table 8: Regression table output for 12 nodes. Edge subsets are shown in the first

columns, and the result taken over all edges is shown in the last column.

Results for 30 nodes:

triadic microstructure	105 edges	102 edges	99 edges	96 edges	all edges
in-stars (021U)	0.00510***	0.00610***	0.00599***	0.00735***	0.00577***
	(7.29)	(16.34)	(14.28)	(11.01)	(32.68)
out-stars (021D)	0.00767***	0.00670***	0.00743***	0.00762***	0.00661***
	(10.15)	(18.04)	(16.69)	(12.68)	(35.56)
_cons	-1.322***	-1.253***	-1.237***	-1.325***	-1.106***
	(-18.46)	(-30.43)	(-30.12)	(-23.36)	(-62.98)
<i>N</i>	64	194	182	114	2000

t statistics in parentheses

* $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$

Table 9: Regression table output for 30 nodes. Edge subsets are shown in the first columns, and the result taken over all edges is shown in the last column.

All nodes, on all edges:

Regression 1: Transitive triads and triadic cycles

triadic microstructure	6 nodes	8 nodes	12 nodes	30 nodes
transitive triads (030T)	0.0700***	0.0452***	0.0268***	0.0154***
	(35.03)	(45.08)	(46.79)	(35.76)
triadic cycles (030C)	-0.237***	-0.162***	-0.117***	-0.0680***
	(-16.80)	(-23.09)	(-34.41)	(-56.25)
_cons	0.262***	0.209***	0.224***	0.237***
	(20.56)	(17.17)	(17.39)	(12.62)
<i>N</i>	1810	1956	1994	2000

t statistics in parentheses

* $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$

Table 10: Regression table summary output for each set of nodes.

Regression 2: In-stars and out-stars

triadic microstructure	6 nodes	8 nodes	12 nodes	30 nodes
in-stars (021U)	0.0998***	0.0615***	0.0238***	0.00577***
	(16.42)	(26.34)	(28.56)	(32.68)
out-stars (021D)	0.109***	0.0609***	0.0225***	0.00661***
	(18.41)	(25.67)	(26.93)	(35.56)
_cons	0.101***	-0.178***	-0.380***	-1.106***
	(5.22)	(-9.78)	(-22.28)	(-62.98)
N	1810	1956	1994	2000

t statistics in parentheses

* $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$

Table 11: Regression table summary output for each set of nodes.

DISCUSSION OF RESULTS

Results related to team network structure

The results we saw from the model significantly matched up to our hypotheses. Transitive triads, in-stars, and out-stars are all positively correlated with the networks formed by our simulation, and triadic cycles were negatively correlated with the networks formed by our simulation.

Results related to the “wisdom of crowds”

An unexpected result from running the model was the very noticeable downward trend in the presence of structures associated with TMSs as the node sizes of networks increased. This tends to suggest that larger teams have a harder time developing the structures associated with higher-productivity TMSs. Recent research by Galesic, Barkoczi, and Katsikopoulos (2016) has found the same result empirically, and offers its own theoretical explanations. Galesic et al defined “moderately sized groups” as groups of about 5 – 30 people, and note that jury sizes, town councils, labor union governing bodies, parliamentary committees, policy boards, and the social network individuals rely on to help them make important decisions rarely exceed 30 members, and tend to stay below 15 members. That is, while making decisions in groups tends to result in more accurate decisions due to the “wisdom of the crowds” effect (Galton, 1907; Condorcet, 1785), empirically people tend to limit the size of the groups in which they make decisions. As summarized in Galesic et al (p. 2), previous explanations of limitations on group size rested on the cost of coordinating larger groups, as in Dunbar (1993).

Galesic et al theorize that smaller group sizes make more accurate decisions. We believe their result is complementary to our result, which shows that microstructures associated with transactive memory systems are relatively more frequent in smaller group sizes compared to random networks of the same size, than in larger group sizes. As these microstructures are related with the storing, making a directory, and retrieving of expertise, the implication is that larger group sizes aren’t able to store, make directories, or retrieve expertise as efficiently as smaller groups.

Attempts and results related to team productivity

Research on transitive memory systems usually assume that better information flow in TMSs indicates higher team productivity (Lee, Bachrach, and Lewis, 2014). However, in real life, teams don't necessarily work efficiently even though they communicate well. Since our model is a "communication network model," meaning that the simulated networks are constructed based on team member communication, we wish to test if efficient communication between members in TMS leads to higher team productivity. We extended the NetLogo model by giving teams a new list of tasks to test how efficient they are. The model didn't give many insights on team productivity, however. We attempted to compare the productivity between TMS networks with random networks by giving TMS networks a new list of tasks. In a mature TMS network, each member knows who to ask when they have too few skills in a certain area to complete a task. That is, there exists a directory of who knows what. In a random network, there's no such awareness of who knows what. Whether we can design a random network with such attributes requires another layer of analysis. This could indicate that this particular productivity measure of a TMS network requires real world data, and that a pure computational model is not enough.

6. CONCLUSION AND FUTURE DIRECTIONS

The simulation of the creation of transactive memory systems presented in this paper is a simple, base-stage model of how transactive memory systems may emerge in

groups of people with varied expertise who need to coordinate in order to complete tasks. Currently, the active agent-node at each step can either complete a task if it has enough expertise, or chain its task expertise with a single other agent on the team. A more realistic model would allow more than up to two agents in a team to complete a single chained task. Time is also not realistically handled in this model. A task-completer can complete any set of tasks in a number of steps equal to the length of the set of tasks. Tasks should take more than one-time step to complete, and if more than one person is getting help from an expert, they should need to wait in a queue for that expert to finish her other tasks first, or to try alternative methods, like going with the next-best expert or to chain experts together. Also, not everyone likes to work hard (at the fastest possible pace); coordinators need to be able to take into account not just who the experts in the team are, but how efficient they need to be in completing a task. In this version, quality of a project outcome may conflict with how many projects can be completed in any given timeframe.

In the future, we plan to address two additional hypotheses in this paper associated with transactive memory systems: 1) that transactive memory systems are measurably more productive than random networks with the same number of nodes and edges (Palazzolo, 2005; Lee, Bachrach, and Lewis, 2014), and 2) that coordinator nodes are more likely to occupy the tertius position of a transitive triad (Lee, Bachrach, and Lewis, 2014). As noted in the results, our initial approach to measuring productivity in our simulated transactive memory systems wasn't successful. As such, we would take a different approach to 1), perhaps by fashioning a test that would see how accurately the TMS can "solve" any given problem compared to a random network of the same size,

along the lines of the arguments for the better accuracy of moderately sized groups in Galesic et al (2016).

REFERENCES

- Anand, V., Manz, C. C., & Glick, W. H. (1998). An Organizational Memory Approach to Information Management. *The Academy of Management Review*, 23(4), 796–809. <http://doi.org/10.2307/259063>
- Argote, L., & Ingram, P. (2000). Knowledge Transfer: A Basis for Competitive Advantage in Firms. *Organizational Behavior and Human Decision Processes*, 82(1), 150–169. <http://doi.org/10.1006/obhd.2000.2893>
- Axtell, R. (2000). Why agents?: on the varied motivations for agent computing in the social sciences.
- Axtell, R. (2005). The complexity of exchange. *The Economic Journal*, 115(504), F193–F210.
- Batagelj, V., & Mrvar, A. (2001). A subquadratic triad census algorithm for large sparse networks with small maximum degree. *Social Networks*, 23(3), 237–243. [http://doi.org/10.1016/S0378-8733\(01\)00035-1](http://doi.org/10.1016/S0378-8733(01)00035-1)
- Borgatti, S. P., & Cross, R. (2003). A Relational View of Information Seeking and Learning in Social Networks. *Management Science*, 49(4), 432–445.
- Condorcet, M. (1785). *Essai sur l'application de l'analyse à la probabilité des décisions rendues à la pluralité des voix* [Essay on the application of analysis to the probability of majority decisions]. Paris, France: Imprimerie Royale.
- Faraj, S., & Sproull, L. (2000). Coordinating Expertise in Software Development Teams. *Manage. Sci.*, 46(12), 1554–1568. <http://doi.org/10.1287/mnsc.46.12.1554.12072>
- Galesic, M., Barkoczi, D., & Katsikopoulos, K. (2016, May 30). Smaller Crowds Outperform Larger Crowds and Individuals in Realistic Task Conditions. *Decision*. Advance online publication. <http://dx.doi.org/10.1037/dec0000059>
- Galton, F. (1907, March 7). Vox populi. *Nature*, 75, 450–451. <http://dx.doi.org/10.1038/075450a0>
- Garner, J. T. (2006). It's Not What You Know: A Transactive Memory Analysis of Knowledge Networks at NASA. *Journal of Technical Writing and Communication*, 36(4), 329–351. <http://doi.org/10.2190/U636-4844-2323-W071>
- Hauke, J. (2014). Organizational routines: an agent based model replication. *Social Simulation Conference*. Retrieved from <http://ddd.uab.cat/record/128522>
- Kitcher, P. (1990), "The Division of Cognitive Labor", *Journal of Philosophy* 87: 5–22.
- Kitcher, P. (1993), *The Advancement of Science*. Oxford: Oxford University Press.
- LeBaron, B., & Tesfatsion, L. (2008). Modeling macroeconomies as open-ended dynamic systems of interacting agents. *The American Economic Review*, 98(2), 246–250.
- Lee, J.-Y., Bachrach, D. G., & Lewis, K. (2014). Social Network Ties, Transactive Memory, and Performance in Groups. *Organization Science*, 25(3), 951–967. <http://doi.org/10.1287/orsc.2013.0884>
- Monge, P. R., & Contractor, N. (2003). *Theories of Communication Networks* (1 edition).

- Oxford ; New York: Oxford University Press.
- Palazzolo, E. T. (2005). Organizing for Information Retrieval in Transactive Memory Systems. *Communication Research*, 32(6), 726–761.
<http://doi.org/10.1177/0093650205281056>
- Palazzolo, E. T., Serb, D. A., She, Y., Su, C., & Contractor, N. S. (2006). Coevolution of Communication and Knowledge Networks in Transactive Memory Systems: Using Computational Models for Theoretical Development. *Communication Theory* (10503293), 16(2), 223–250. <http://doi.org/10.1111/j.1468-2885.2006.00269.x>
- Peltokorpi, V. (2008). Transactive memory systems. *Review of General Psychology*, 12(4), 378–394. <http://doi.org/10.1037/1089-2680.12.4.378>
- Qian Huang, Hefu Liu, & Xuepan Zhong. (2013). The impact of transactive memory systems on team performance. *Information Technology & People*, 26(2), 191–212.
<http://doi.org/10.1108/ITP-04-2013-0068>
- Rulke, D. L., & Galaskiewicz, J. (2000). Distribution of Knowledge, Group Network Structure, and Group Performance. *Manage. Sci.*, 46(5), 612–625.
<http://doi.org/10.1287/mnsc.46.5.612.12052>
- Shore, J., Bernstein, E., & Lazer, D. (2015). Facts and Figuring: An Experimental Investigation of Network Structure and Performance in Information and Solution Spaces. *Organization Science*, 26(5), 1432–1446.
<http://doi.org/10.1287/orsc.2015.0980>
- Stevens, M. (2003), “The Role of the Priority Rule in Science”, *Journal of Philosophy* 100:55–79.
- Su, C. (2012). Who Knows Who Knows What in the Group? The Effects of Communication Network Centralities, Use of Digital Knowledge Repositories, and Work Remoteness on Organizational Members’ Accuracy in Expertise Recognition. *Communication Research*, 39(5), 614–640.
<http://doi.org/10.1177/0093650211433825>
- Wegner, D. M. (1995). A Computer Network Model of Human Transactive Memory. *Social Cognition*, 13(3), 319–339. <http://doi.org/10.1521/soco.1995.13.3.319>
- Yuan, Y. C., Carboni, I., & Ehrlich, K. (2010). The impact of awareness and accessibility on expertise retrieval: A multilevel network perspective. *Journal of the American Society for Information Science & Technology*, 61(4), 700–714.
<http://doi.org/10.1002/asi.21287>
- Yuan, Y. C., Fulk, J., Monge, P. R., & Contractor, N. (2010). Expertise Directory Development, Shared Task Interdependence, and Strength of Communication Network Ties as Multilevel Predictors of Expertise Exchange in Transactive Memory Work Groups. *Communication Research*, 37(1), 20–47.
<http://doi.org/10.1177/0093650209351469>