7-7-2020

Modeling the Evolution of Agent Capabilities and Specialization

Radhika Jayaraman
University of Windsor

Modeling the Evolution of Agent Capabilities and Specialization

By

RADHIKA JAYARAMAN

A THESIS

Submitted to the Faculty of Graduate Studies

Through Computer Science

In Partial Fulfilment of the Requirements for

The Degree of Master of Science at the

University of Windsor

Windsor, Ontario, Canada

2020

Modeling the Evolution of Agent Capabilities and Specialization

By

RADHIKA JAYARAMAN

Approved by:

April 28, 2020

Declaration of Co-Authorship

I. Co-Authorship

I hereby certify that this thesis incorporates material that is the result of research conducted under the supervision of Dr. Ziad Kobti. In all cases, the model framework, experimental design, and analysis, interpretation and writing are contributed solely by the author. The contribution of the co-author was primarily through the proofreading of the manuscripts.

I am aware of the University of Windsor Senate Policy on Authorship and I certify that I have properly acknowledged the contribution and have obtained permission from the co-author to include the above material(s) in this thesis.

II. General

Some of the material in this thesis is inspired by the work found in thesis 1 [1], thesis 2 [2] and thesis 3 [3] for setting a framework foundation for the model and understanding the different techniques involving the evolution of agent capabilities and adaptation of specialization techniques.

I certify that, with the above qualification, this thesis, and the research to which it refers, is the product of my own work. I certify that, to the best of my knowledge, my thesis does not infringe upon anyone’s copyright nor violate any proprietary rights and that any ideas, techniques, quotations, or any other material from the work of other people included in my thesis, published or otherwise, are fully acknowledged in accordance with the standard referencing practices.

I declare that this is a true copy of my thesis, including any final revisions, as approved by my thesis committee and the graduate studies office and that this thesis has not been submitted for a higher degree of any other University or Institution.

Abstract

A social system is a patterned network of interrelationships that exist between individuals, institutions, and groups forming a coherent whole. Understanding the varying system outcomes for different decision-making processes selected under varying environment constraints in advance will aid in the realization the of best decision towards an effective outcome. One of the ways to increase system productivity is ‘Agent Specialization’. Also, the agents (individuals) who operate as generalists are most vulnerable to being replaced. Therefore, there is a need to focus on agent specialization to enhance the ability of an agent along with the evolution of an agent. Multi-Agent Based Simulation, a subfield of distributed AI, provides a technique to naturally describe a social system. To help improve decision-making intricacies of the agents to evolve and specialize, there is an increasing need to formulate an enhanced model of MABS. This thesis proposes a novel framework that exploits the benefits of social networks providing a decision support system for agent (individual) specialization by integrating the concept of ‘Positive Social Influence’ exerted by experts in the system. Consequently, the proposed framework assists the growth of agents by enabling the evolution of agent capabilities with the identification of suitable producer-agents using an evolutionary component (cultural algorithms). Enabling agent specialization and assisting the ability of the agents through capability evolution is anticipated to increase the productivity of the system. Evaluation of results shows the successful evolution of agent capabilities with the identification of suitable producer-agents in an optimized aspect (reduced operational cost and reduced distance cost) in comparison with exhaustive search, random search, and genetic algorithms and the improved degree of specialization of agents (increased dol values with a minimum of 3% increase to a maximum of 16.7% increase in comparison with standard genetic threshold model for varying agents and task number) in a given dynamic environment.

Dedication

I dedicate my work to my beloved mother *Mrs. Uma Maheshwari* and my loving Friends, for their continuous support and understanding. I am glad to have found selfless well-wishers as friends and I will forever be grateful for their motivation and friendship. Thank you all for feeding me guidance and positivity during the times of my struggle.

Acknowledgement

I would like to grab the opportunity to pay gratitude to my supervisor, *Dr. Ziad Kobti* for his continuous support and constant understanding. I am humbled by his immense support by contributing his suggestions and ideas throughout my research duration. His positive attitude and insightful feedback have helped me in great ways.

I would like to express my appreciation to the Ex. Graduate Secretary of the University of Windsor, School of Computer Science, *Ms. Karen Bourdeau* for her constant support. I want to extend my sincere thanks to *Ms. Melissa Robinet, Ms. Christine Weisner, Ms. Gloria Mensah* and *Ms. Tina Palmer* who always supported me when I needed assistance in various academic issues.

I want to express my gratitude to *my family* and *friends* for their unconditional love and positivity.

Table of Contents

Table of Figures

Table of Tables

Abbreviations/Symbols

AI:	Artificial Intelligence
MABS:	Multi Agent Based Modeling
MAS:	Multi Agent Systems
ABM :	Agent Based Models
SN:	Social Networks
GA:	Genetic Algorithm
CA:	Cultural Algorithm
BS:	Belief State
TS:	Task Selection
PSI:	Positive Social Influence
DSS:	Decision Support System
$\triangleq$:	is defined as
$\in$:	Element of
$\bigcup$:	Union of
$\sum$:	Summation of
DOL:	Division of Labor

Chapter 1: Introduction

A social system is a patterned network of interrelationships that exist between individuals, institutions, and groups forming a coherent whole [30]. Examples of social systems include societies, family groups, cities, and corporations. Gaining insight into the operations of the system, understanding the dynamic behavior of the system in response to the existing conditional parameters and predicting the outcomes of such social systems will help in formulating the systems prominently. The AI (Artificial Intelligence) tools and technological resources like DSS (Decision Support Systems) and MABS (Multi-Agent Based Simulation) provide a framework to mimic and infer the outcomes of the social systems, in turn aiding the effective formulation of prominent systems.

MABS is a powerful simulation modeling technique that has distinguished itself over the past two decades in naturally simulating the system by providing an understanding of intricacies involved in the evolution of systems, modeling the communication between the actors in the system, enhancing learning abilities of the actors in the system, etc.

This thesis directs the work in the direction of more advanced work done using MABS as a tool to involve the SN (Social Networks) and CA (Cultural Algorithm) components to formulate a social system and to infer inductive and near-optimal outcomes of the given social system as precisely as possible by considering various varying parameters (constraints / existing conditions) in a dynamic environment. Analyzing the outcomes of the mimicked social system subjected to varying constraints helps in determining the most feasible decision-making process that would improve the overall efficiency/productivity of the system. Therefore, the proposed MABS model assists the entities of a social system in electing a feasible decision-making process, in turn, resulting in the efficacious system.

As discussed, a social system is a complex structure consisting of numerous actors, interacting with each other in different ways in a given environment. The evolution of individuals (growth of individuals over time) in a social system, the evolution of a social system on whole, and the social interactions in such systems are to be observed in order to understand and analyze the outcomes of the given system. Imitation and decision-making in such a complex system is a tough task as it involves a complicated analytic process. Multi-Agent Based Simulation (MABS), provides a technology to naturally simulate the evolution of social systems. Multi-Agent Systems is composed of an environment that includes numerous agents with decision-making heuristics, learning, and interacting abilities. MABS helps in simulating various types of scenarios and assist the process of decision-making [4] [5]. Inclusive of the evolution of individuals under the aid of decision-making abilities of MABS, 'Task Characterization' (Task Selection) plays an important part in conceptual modeling and analysis of multi-agent based models. The agents with no control of their tasks can irrationally select or pursue unrealistic tasks. Therefore, task selection is one of the most important aspects of successful agent development [9] [10] [11] [12]. With numerous methodologies for task selection,[3] [13] [14] [15] shows the importance of selecting realistic/feasible tasks.

1.1 Introduction to Multi-Agent Based Simulation (MABS)

Definition- Multi-Agent Based Simulation (MABS): Multi-Agent Based Simulation is a subfield of distributed AI, referring to the simulation technique that infers the behavior of a complex system from the actions and interactions of the individuals in the system. Understanding individual behavior and multiple interactions amongst the individuals in the given system, MABS can flexibly capture the emergent phenomena of the system.

MABS system is modeled as a collection of autonomous decision-making entities called agents. Each agent executes various roles with relevance to the system (Example: Producer, Consumer/Seller). At the basic level, the MABS model is comprised of agents and the relationships between them.

MABS systems are capable of naturally describing the social systems and generate observable emergent behavior at the population level [6] [7]. Besides, they help to explore the system in a controlled environment aiding experts to observe the behavior of the system with ease [8]. MABS mimics the social systems and the task set comprising of all the tasks available in the system, subjective to the dynamic environment is gathered. A complex task in the system can be divided into smaller tasks and each of them will be assigned to an intelligent agent comprising MABS, for the system to be implemented and explored in detail [8]. The intelligent agents [16] composing the system has a particular responsibility and have a particular role such as a producer or a consumer in a given environment. Also, an intelligent agent is defined as a problem-solving entity that works autonomously in an environment to achieve its task. That is, multiple interacting intelligent agents in Multi-agent systems can solve problems that are difficult for an individual agent to solve.

MABS applications have transitioned from the modeling of simpler societies to complex human societies.

1.2 Introduction to Social Networks (SN)

Definition- Social Networks: Social Network is defined as a network of social actors (such as friends, individuals, colleagues, and acquaintances) connected by interpersonal relationships. Social Networks are the best way to model interactions amongst individual entities in a social system [41]. With small-world networks best mimicking the real-world networks, the inclusion of

SN component in MABS helps in enhancing the model by capturing the digital connectivity of the social system, in turn aiding the evolution of the system.

1.3 Introduction to Agent Specialization

Definition- Agent Specialization: Specialization, concerning context, means perfecting (specializing) one task rather than generalizing in multiple tasks towards the productive output. Agent Specialization in simple terms, allows the agents to focus on one task rather than focusing on many which in turn enables the agents to become more adept at a specialized task, increasing efficiency and productivity on an individual (agent) and population level (system/corporation).

Specialization is one way in which agents can increase their productivity [20] by co-operating with other individuals. One such way of increasing productivity can be achieved by the division of labor technique. The study of specialization is important in several fields like Archaeology, Biology, etc. Archaeologists study specialization to understand the changes in societies. In biology, specialization helps to understand the behavior of biological creatures such as wasps and ants [13] [14] which shows specialization based on tasks. Many of these fields use computer simulations to study the effect of specialization on productivity. Specializing in a subset of tasks available within a system allows agents to more efficiently fulfill system demands. In dynamic environments, considering each task having unlimited resource supply and limited demand, 'Threshold Reinforcement' is the more suitable approach to Specialization [21]. Another research [22] considering dynamic environments, shows the emergence of specialization by accounting the stimuli of global task demands. In [22], the parameter of environment, global demand, is considered rather considering the parameters of interactions amongst the fellow agents in the system.

[3] explores the idea of positive social influence leading to agent specialization. Every agent possesses a genetic threshold for available tasks. When the thresholds for multiple tasks has been surpassed, agents will consider the choices of their neighbors in choosing which task to perform. Standard ‘Genetic Threshold Model’ selects the available tasks randomly, whereas ‘Positive Social Influence Model’ [3] selects the tasks influenced by the neighbor agents. [3] shows the improved levels of specialization under social influence.

1.4 Introduction to Cultural Algorithms (CA)

Definition- Cultural Algorithms: Cultural Algorithms (CA) are computational models of cultural evolution [46] [47] and comprises of population and belief space connected via a communication protocol. In brief, CA is described as a class of optimization algorithms and is inspired by ‘Evolution’. Evolution in nature is optimizing just about everything over billions of years to maintain order by reinforcing the ‘survival of fittest’ conception. With this background, computer scientists developed optimization tools known as evolutionary algorithms. One such evolutionary algorithm is CA. The two main components of CA include population space and belief space. Population Space is comprised of chromosomes that are filled with genes, meaning, the candidate solutions to the given problem are gathered/generated in population space. Belief Space is used as a knowledge repository to maintain the best candidate solutions of the population. The initial population has chromosomes/candidate solutions, a pair of chromosomes from the population is selected and is termed as parents. The parent chromosomes mate and create new chromosomes termed as children/offsprings through crossover events (mix and matches the genes from parent chromosomes) and mutation (altering one or more gene value of the chromosomes). This leads to genetic variations and small-scale adaptations. If the adaptions create individuals (offsprings) better suited for the environment (survival of the fittest), then such individuals are added to the population space. Individuals that are poorly suited to the environment are discarded. That is, the fitness of the offsprings created is calculated using a fitness function, that takes the candidate solutions/offsprings/individuals of the problem as input and assigns a fitness score to each individual by determining how fit the created solution is. The best solutions are added to the population (selection) and the poor solutions are discarded. The hand-picked best solutions are updated to belief space. Now, for the next generation, parents from the population and the influence of belief space are considered for the creation of offsprings. The same process is repeated over generations. Over a large number of generations, certain genetic adaptations become fixed in the population. In this way, the algorithm optimizes a population best suited for the given problem.

In papers [64][65] the evolutionary learning process in the context of optimization is discussed. The authors of [66] show the amalgamation of learning algorithms in multi agent-based modeling. The paper shows the importance of evolutionary algorithms in decision support systems, mainly in dynamic environments.

In this thesis, a social system is mimicked using MABS. The MABS framework comprises of a dynamic environment with varying constraints and multiple agents interacting with one another. Modeling the communication between the social agents in a complex system is realized with the help of social networks [23] [24]. The individuals of the complex social system are represented using intelligent agents. An intelligent agent can be defined as a problem-solving entity that is working autonomously in the environment to achieve its destined goal. The agents in the system are classified into two roles: Producer-Agents and Consumer-Agents and are mapped using a social network that defines the relationship between the social agents (individuals) in a population.

Producer-Agents are the agents in the network that exerts influence and provides services to the Consumer-Agents. The term experts, influencers and Producer-Agents might be used interchangeably in this work. Whereas, Consumer-Agents are the agents in the network that are influenced by experts/Producer-Agents and consume the services from Producer-Agents to evolve/grow. To begin with, the Consumer-Agent needs to select a task from Task-set which comprises of all tasks within the system/environment. Task selection using the 'Positive Social Influence' factor will result in the creation of specialized agents. The positive social influence in selecting a particular task in exerted by the experts who are directly connected to the Consumer-Agents. Spending relatively more resources/time on performing the selected task rather than spending an equal amount of resources/time on all tasks results in the emergence of specialized agents. Once the task is selected, the Consumer-Agents in the network must have a required set of capabilities to achieve its selected task.

With capabilities referring to the agent's task-directed subset of requirements, at initialization, every Consumer-Agent has minimal capabilities. Every so often, Consumer-Agents do not possess all the required capabilities to achieve their tasks. Therefore, the Consumer-Agents has to rely on the services/capabilities offered by Producer-Agents in the network, to acquire the required set of capabilities. As the inclusion of the cultural evolutionary component in MABS results in a powerful tool for modeling [25] [26] [27] [28], a Cultural Algorithm component is infused in the framework to aid the identification of best/suitable Producer-Agents in the network. Though there exist proposed models of MABS on modeling the evolution of agent capabilities [1] [2] [29] and models to formulate the emergence of agent specialization [3] [18] [19], with plenty of room to advance, there is a need for an enhanced model to synthesize the growth of 'Agent Capabilities' and improve the degree of 'Agent Specialization'. In turn, assisting the agents, where the agents are aided not only to grow but also to grow into specialized agents in a dynamic environment.

Conclusively, in this thesis, a novel MABS model is proposed in which the main social actors of the system including Producer-Agents and Consumer-Agents are mapped using a 'Social Network' in order to realize the agent interactions. And the Consumer-Agents selects the desired task by considering the ‘Positive Social Influence’ factor exerted by experts in an environment resulting in ‘Agent Specialization’. Correspondingly, a CA (Cultural Algorithm) component is included to aid the identification of the suitable Producer-Agents with respect to the capability needs of Consumer-Agents. The resulting model synthesizes the intellection of ‘Capability Evolution’ and ‘Agent Specialization’ thus assisting the agents to evolve and specialize in the given dynamic environment.

1.5 Problem statement

In the social system under study, the environment comprises of the Task-set, Consumer-Agents, and Producer-Agents as main actors. Task-set is the collective representation of tasks available to perform in a given environment and each task is defined as, $t \in \mathrm{TS} \triangleq (\mathrm{tid}, \mathrm{S}, \mathrm{ReqC})$ where TS denotes the Task-set, tid denotes the task id, S denotes the stimulus intensity and ReqC represents the required capabilities to perform the task and is defined as, $ReqC = (reqc1, reqc2, reqc3 \ldots . reqcn)$ where reqc1, reqc2, reqc3 till reqcn represents the abilities required by a Consumer-Agents to achieve the selected task (Example, reqc1: Reading, reqc2: Writing, reqc3: Coding, reqc4: Problem-solving ability, reqc5: Technical skills, reqc6: Research skills, etc). Consumer-Agents are characterized as the consumer class agents of the system and each Consumer-Agent is defined as, $ag^c \in C_{AG} \triangleq (Cag^c, \mathrm{T}_T)$ where C_{AG} represents Consumer-Agent class, T_T denotes the tasks and the corresponding threshold available for each task. That is, for every task in the environment, an internal threshold is assigned. T_T is defined as, $T_T \in (\theta T_j)$ where θT_j is the threshold associated with task T_j and the value j represents the task id of every task in the environment. Cag^c represents the capability set of the Consumer-Agent and is defined as, $Cag^c = \{c1|c2 \ldots . |cn\}$ where c1, c2 till cn represents the abilities possessed by the

Consumer-Agents (Example, c1: Reading, c2: Writing, c3: Problem-solving ability, etc). Producer-Agents are characterized as the producer/provider/expert/influencer class agents of the system and each Producer-Agent is defined as, $ag^{p} \in P_{AG} \triangleq (Pag^{c}, EX)$ where P_{AG} represents Producer-Agent class and Pag^{c} represents the capability set offered by the Producer-Agent and is defined as,

$Pag^{c} = \{c3, op | c9, op \ldots . | cn, op\}$ where c3, c9 till cn represents the capabilities/services offered by the Producer-Agents. (Example, c3: coding services, c9: Typing services, etc) and 'op' associated with every capability/service denotes the operational cost/service charge of the capability provided/serviced. *EX* represents the expertise of the Producer-Agent.

Consequently, every Consumer-Agent selects a task subjected to varying constraints in the environment and has to have respected required capabilities to achieve its selected task but is not the case every time. Because the Consumer-Agents has minimal capabilities at initialization and hence has to rely on the Producer-Agents to acquire the required capabilities. Multiple Producer-Agents may be needed to support a Consumer-Agent.

Now, the problem is to select a task on which relatively more resource/ time is to be spent in order to specialize in the selected task and to identify suitable Producer-Agents to rely on. (Example: Supposedly, Consumer-Agent selects 'Computer Engineering' as a task from the Task-set and the capabilities including 'reading, writing, coding, problem-solving ability, technical skills, and research skills, etc.' are the required capabilities to achieve the task. But initially, the Consumer-Agent has minimal capabilities and the capabilities like coding, technical skills, and research skills are considered to be missing from the agent's capability set. Now, with multiple Producer-Agents in the system offering the required missing capabilities, how to identify suitable Producer-Agents?).

That is, how can Consumer-Agents select suited tasks in a dynamic environment to attain specialization and evolve by acquiring required capabilities from suitable Producer-Agents during the process to achieve its respective selected tasks?

The agents (individuals) who operate as generalists are most vulnerable to being replaced and with a need to focus on agent specialization (Example: how to specialize in Computer Engineering), the primary research question is formalized as, ***how can Consumer-Agents evolve in a dynamic environment by selecting feasible tasks to perform and acquiring required capabilities from suitable Producer-Agents during the process to attain specialization and achieve its respective tasks?***

1.6 Thesis statement

The social system under study has Task-set, Consumer-Agents, and Producer-Agents. We map the Consumer-Agents and Producer-Agents to the nodes of the social network (SN). The social network representing the system is an undirected weighted graph with nodes representing agents and edges are associated with distance cost. The distance cost can be the travel time taken or the resource required to travel from one node to another. The efficiency of the social system is enhanced by increasing the productivity of the system using the specialization concept. Improving the degree of specialization and assisting the ability of the agents through capability evolution with identifying suitable Producer-Agents in the system is anticipated to increase the productivity of the system.

If we introduce Producer-Agents in the social network and consider the positive social influence exerted by them in selecting the tasks, then we expect to see the increase in the degree of specialization. The degree of specialization in measured using the division of labor (dol) technique. Our method is compared with the standard genetic threshold model where the agents select their tasks at the absence of experts. That is, the agents in the genetic threshold model select the available tasks randomly without considering the influence of the experts.

Furthermore, if we use the evolutionary algorithm (CA) in identifying suitable Producer-Agents to assist the required capability evolution of the Consumer-Agents then we expect to identify the team of Producer-Agents providing the required capabilities/services at (near-optimal) minimum operational cost and (near-optimal) minimum distance cost at reduced run time. We measure the performance of the algorithm by calculating the average operational cost and average distance cost of the identified team of Producer-Agents. The run time of the algorithm is evaluated (algorithm execution time in seconds). Our proposed algorithm is compared with exhaustive search, random search, and genetic algorithm.

1.7 Background

1.7.1 Social Systems

A social system can be described as a group of social actors with common goals. In any given environment, the social actors are interacting actors with basic orientations [30]. The examples of social systems include a group of friends, neighbors, a circle of family, etc., which plays a key role in everyday human life. The interactions amongst the actors of the social systems and the social system on a whole critically impact the society and the people(individuals) in society. Hence, they are studied from both individual and systematic perspective. A social system and the fundamental foundations of such interacting systems can help the experts to systematize the society to study the behavior and interactions amongst the social actors of the system.

1.7.2 MABS

Multi-Agent Based Simulation (MABS) is a well-established discipline for modeling social systems. A social system can be described as a group of social actors with a patterned network of relationships between them [30]. Examples of social systems include a corporation, a government, society, family, friends circle, etc. The individual actors/agents are entrusted with a task or responsibility in order to perform a collective function or a common goal. The individuals are interdependent as they form a part of the social system. The individuals/actors in a social system can make choices that affect the state of their environment. Social systems aid experts to classify society into social systems to learn the behavior and interactions amongst them [30]. Due to the critical impact of social systems on people and society, they are to be explored both from an individual perspective and a systematic perspective.

Multi-Agent Systems (MAS) or Agent-Based Models (ABM) are beneficial with environments that comprise complex interactions, possess dynamic population size or exhibits complex agent's behavior including learning ability [31]. Agent-based models are composed of multiple interacting agents operating in an artificial world which in turn helps imitate the social systems for better understanding. The interacting autonomous agents of ABMs can represent humans, organizations or any entity that can act upon its environment. ABM uses the power of computers to explore the dynamics of the given system which is out of reach for pure mathematical methods. This characterization and several other characteristics of ABMs makes them a prominent technology for modeling social systems. The models are considered flexible, as the autonomous interacting agents of the model can be added or removed as per requirement. Also, the agents can be created with a changing degree of rationality and can be given the ability to learn in the presence of other agents [31]. Applications of Agent-Based Models for analyzing social systems span over a variety of fields including social science, engineering, economics, biological science, etc, [7]. Additionally, the widespread interest in ABMs is that they as a technology, aid in the development of new models in human-centered systems. This very aspect of ABM allows people with no technical background to interact with the model in a simple way [32].

Social simulation is a scientific discipline concerning social interactions in the system. Simulating social interactions in social systems helps in studying various issues in social sciences. Multi-Agent Based Simulations (MABS) also referred to as Multi-Agent Based Social Simulations are social simulations built using ABMs. In MABS, the goal of ABMs is to examine informative insight into the collective behavior of agents obeying simple rules, typically in natural systems, rather than in designing agents or solving specific practical problems [33]. The modeling process of MABS is best described as inductive. That is, the inputs/evidence given to the model results in a probable conclusion rather than a certain conclusion. This inductive quality of MABS helps the modeler to articulate assumptions that are relevant to the dynamics of circumstances and then watch the assumptions that are relevant to the dynamics of circumstances and then watch the phenomena emerge from the agent’s interactions [34]. Modeling the communication/interactions between the social agents in a system can be realized with the help of social networks [23] [24]. Additionally, the inclusion of the cultural evolutionary component in MABS systems results in a powerful tool for modeling the system [25] [26] [27] [28].

1.7.3 Agents

In computing, the term “agent” implies an entity that performs a task or collective task on behalf of one or more humans/individuals. Also, any agent that can observe and act upon its environment is termed as an intelligent agent [36]. In [35] the authors describe different types of agents in the field of Artificial Intelligence (AI): Simple, Problem-Solving and Learning Agent. A simple agent is the one which receives data from the environment and reacts to the environment. A simple agent receives data about the current situation of the environment, matches the same with the rules of the agent and selects the best action accordingly. Whereas, a problem-solving agent decides to choose a proper action based on environmental situations and the agent’s goals. In other words, it measures the effect of its acts on the environment to find out how the series of actions can lead it to the acceptable states (goal). A learning agent is yet another type of agent which stores the obtained knowledge of its previous experiences and use them in the future action selection process. Therefore, an intelligent agent can be defined as a problem-solving entity that is working autonomously in the environment to achieve its destined goal. Also, an intelligent agent is adaptable to the environment and receives the data from the environment by sensors and react to the environment by actuators [37][8][35].

1.7.4 Agent Capabilities

The concept of capability was introduced in the BDI (Belief-Desire-Intention) model [38] to make the agent more adaptable. The agent acts as an entity in an environment and adding flexibility to the agent's characteristic enables it to adapt to the dynamics of its environment. Capabilities, at the foundation level, represent the task-directed/goal-directed subset of requirements that an agent must possess in order to achieve its tasks/goals. In [39] the authors defined and formulated the capability in a single BDI agent. In [40], the authors extend the definition of capabilities into two classes: Internal capabilities and External capabilities, to facilitate the agent's collaboration with other tools or agents. Internal capabilities are formalized as those capabilities that the agent already has. As internal capabilities alone are not enough for an agent to achieve its tasks, there is a need for the agents to acquire external capabilities. The agents can acquire the capabilities to achieve their tasks by various techniques including evolutionary computation algorithms [2]. That is, the agent capabilities can be synthesized to evolve over-time using evolutionary algorithms like CA to achieve its tasks.

1.7.5 Social Networks

Social Networks can simply be described as a set of social actors connected with a set of relationships between them. Social Networks are the best way to model interactions amongst people in a community. An ideal way of representing these networks is by mapping them to a graph structure [41]. A social network can be represented using: Unweighted graph (or) Weighted graph. G = {V, E} represents an unweighted graph where "V" is the set of vertices (actors-can be a person, organization or any other entity) and "E" is the set of edges (relations). Whereas, in a weighted graph, the edges are associated with some weight "w". The weight associated is usually decided based on the similarity of two nodes, the distance between them or frequent relationships amongst them. This weight usually is a numeric value that represents how closely the social actors/vertices are tied together. In addition to the weighted and unweighted characteristics, a graph can either be directed or undirected as well. A directed graph comprises a set of vertices connected by edges, and the edges have a direction associated with them [42] as opposed to the undirected graphs [43]. Such graphs with n vertices can be described by an adjacency matrix (n by n matrix). The value in the matrix is either 1 or 0 based on the presence or absence of the link between the respective vertices.

Another important characteristic of the social network is that it displays dynamic behavior, meaning, their topology changes over time. A social network defines a relationship between social individuals in a population. When viewed as a graph the individuals can be seen as nodes with ties between them. Nodes in social networks are characterized by their degree of connectivity and clustering coefficient [44]. Here, node's degree of connectivity is the number of links or connections it has with other nodes, whereas, the clustering coefficient is referred to as the density of which extent one's neighbors are neighbors of each other. With different types of social network models used in agent-based modeling, small-world networks are more of a replica to real-world networks. In small-world networks, the majority of nodes are connected to their nearest neighbors. The small-world phenomenon is the one in which the social actors are linked together through short chains of intermediate friends. That is, the small world phenomenon demonstrates that all the actors in a network are linked via short paths of acquaintances. In addition, the degree distribution in these networks follows the power-law distribution [45]. The power-law pattern is very common in real-world networks and SN (social network) plays the main role in the evolution of social systems mainly because of its digital connectivity.

1.7.6 Cultural Algorithm (CA)

Cultural Algorithms (CA) are computational models of cultural evolution [46] [47] and comprises of population and belief space connected via a communication protocol. Belief space gathers and maintains the experiences contributed by selected individuals from the evolving population through an acceptance function. The experiences are stored as categories of knowledge sources, which in turn is used to influence the evolution of individuals in the population space through an influence function. Knowledge sources in belief space component are characterized into five categories including situational, normative, topographic, historical or temporal and domain knowledge. Situational knowledge captures the best performers in the population. Normative knowledge maintains encouraging variable ranges and can help individuals leap into good ranges. Topographical knowledge refers to the spatial characteristics of the search space. Historical or temporal knowledge constitutes important events or temporal patterns during the search process. Domain knowledge is knowledge specific to the domain of the problem being addressed by the CA. The knowledge sources can be used selectively or collectively to guide the search process of the CA. The CA framework facilitates extracting, storing and exploiting experiences in a population of individuals over time thus permitting self-adaptation and learning at various levels [48].

The very idea of cultural algorithms is inspired by the natural cultural evolution process. It assumes that by using the knowledge, generations can evolve faster than the regular biological evolution (Genetic Algorithm). CAs have been applied to solve a variety of optimization problems.

1.8 Research Motivation

Considering a social system with multiple interacting social actors advancing towards common goals, foreseeing the outcomes of the system under varying environmental constraints would help in understanding every inductive decision-making process involved. Realization of all such possible decision-making processes under varying constraints in advance will aid in selecting the best decision towards an effective outcome. We require frameworks/models to replicate/mimic the given system at hand. This can be achieved efficiently by incorporating the techniques of AI to learn the possible outcomes of the systems, meaning, people's/individual's decision-making process could be eased and paced with the help of available technological resources like DSS (Decision Support Systems) and MABS (Multi-Agent Based Simulation). MABS systems are built with the objective of understanding the intricacies involved in the evolution of complex systems(societies). A social system consists of linked social circles of services, an environment with certain conditions and constraints, and a growing population(individuals). The individuals of the population who operate as generalists are the most vulnerable to being replaced. Hence, there is a need for the individuals comprising the population to specialize. In order to specialize, the individuals should select a suitable task to spend relatively more resources/time and also acquiring the required capabilities to achieve the tasks becomes important, as borrowing or extension of capabilities through other agents is not ideal always. The fact is, very few researches have explored the system from the social network and with the inclusion of an evolutionary concept perspective.

The idea and the motivation are to offer a better understanding of the decision-making etiquette of individuals in an environment that is subject to social influence. The individuals acquire their required abilities from providers to enhance themselves and evolve with the growing population. Exploring the system from a social network perspective and the inclusion of CA and positive social influence will result in an enhanced model where the individuals(agents) can evolve over-time into specialized agents.

1.9 Research Contributions

An enhanced MABS model, synthesizing agent specialization and capability evolution is proposed in this thesis work. A novel method is introduced to enhance/improve agent specialization by adopting 'Positive Social Influence' factor exerted by experts/influencers/producer-agents realized through social networks and to identify the suitable producer-agents in the network to acquire the agent's capabilities using the benefits of social networks and a CA component. To the best of our knowledge, this is the first attempt to blend the component of evolutionary algorithm and agent specialization techniques. The output of this research will be an augmented/enhanced model that can be used as a decision support system to achieve Consumer-Agent specialization by selecting tasks based on the positive social influence exerted by experts in the network/system and to recommend suitable Producer-Agents in the network to Consumer-Agents, aiding the Consumer-Agents to acquire the required capabilities at minimized operational cost and distance cost.

1.10 Research Objective

Our first objective is to describe the system, its dynamic environment and all the components of the social system in an appropriate computable form using a MABS model. The next objective is to include the ‘Positive Social Influence’ factor exerted by experts in the network which improves agent specialization and to propose an algorithm (CA) that aids in finding suitable Producer-Agents in the environment to assist the Consumer-Agents. The proposed system can provide an opportunity to augment a MABS model by using the benefits of social networks and a CA to achieve the following goals,

- Assisting in a better understanding of environmental dynamics and complexity, in turn assisting agent specialization.
- Finding/Selecting/Identifying the suitable Producer-Agents to provide services.
- Reducing the operational cost and distance cost in acquiring the missing capabilities required to achieve a task.

- Improving the degree of agent specialization.

1.11 Thesis Outline

The rest of this thesis is organized as follows. In chapter 2, some of the approaches and research works that are related to our proposed model are reviewed. Our proposed model and algorithm, the use of knowledge sources in belief space and the role of positive social influence in agent specialization are discussed and explained in detail in chapter 3. The implementation, experimental setup and parameters used in the evaluation of our model are discussed in chapter 4. Evaluation of the obtained results is presented in chapter 5. Chapter 6 presents a discussion of the example system, challenges and limitations that needs to be addressed while implementing our proposed system in real world. Finally, the conclusion and future works are discussed, in chapter 7.

Chapter 2: A Literature review

This chapter fixates on the discussion concerning the existing work that has been built by researchers over the years. Various approaches starting from the eminence of task selection and resource allocation in MABS that results in agent specialization to different techniques aiming evolution of capabilities, signification on the inclusion of SNs in MABS, the usefulness of CA in MABS modeling is discussed.

2.1 Social Influence, Task Selection and Agent Specialization

In reference to the context of the problem we are trying to solve, efficient task selection/allocation amongst the agents become crucial. There are multiple ways to handle task selection problems ranging from simplex algorithms to more efficient algorithms.

The role of social influence in task selection, in turn, resulting in specialization of agents is discussed in detail in this section. Given a set of agents and a set of tasks, it would not be ideal for the agents to select a random task to perform. In [53], a genetic algorithm is proposed to deal with the task selection problem. The functionality of the algorithm is to get the requirements of each task and try to identify the best set of agents to perform that particular task. Genetic algorithms aid in finding the near-optimal solution. It is claimed in [53] that their proposed algorithm can find a near-optimal solution for this way of task selection with stability, robustness, scalability, and accuracy. A task can be done by a single agent or can be divided amongst a team of agents, also, the allocation of a task to a group of agents is decided with respect to the best collaboration rate. In [54], the task selection/allocation problem in a dynamic environment is discussed. In a given dynamic environment, each task can be assigned to an agent or a group of agents. The authors proposed three algorithms for task allocation with the inclusion of the concept of synergy to identify the best group of agents for a given task. The tasks are interdependent, meaning, each task has its priority and many prerequisites. That is if a provider has all the capabilities to perform a task, only then the task would be allocated to it. In case if none of the providers can provide a task, then the task would be divided into sub-tasks. Now the model will look for a team of providers for the sub-tasks. In [1], the authors have explored the system from a social network perspective and have claimed that applying social network techniques on the system can enhance both planning and task selection/allocation processes. In [18], the authors involved the idea of demand levels, which in turn helps in observing the behavior of their proposed approach when there is a variation of demands for a task. Demand is the total amount of effort needed to satisfy all tasks relative to the total work ability of all agents. The variation of demand levels ranges from a shortage of demand for a task to an excess of demand for a task. It is shown that the concept of positive social influence in task selection was able to increase the level of task specialization within a population given all demand levels. In [55], the authors propose a model to show the importance of social influence in impacting the decisions of the agents with the help of social networks. The model accounts the local interactions amongst the agents within the network and the impact of interactions (impact of social influence) in formalizing the decision (opinion/attitude) of the agents. Selecting a task and spending the agent resource on one selected task results in the emergence of specialization in agents. The use of DOL (Division of Labor) technique in beehives, aiding the adaptive mechanism of its colony in discussed in [56]. During winter, the goal of the beehive is survival, and hence the workers are all generalists. Whereas, in summer, the goal of the beehive changes to maximize growth and accumulation. Group of workers are divided to perform particular tasks eventuating specialization.

Specialization is one way in which agents can increase their productivity [20] by co-operating with other individuals. There are several approaches to model agent specialization, few of them includes, Effect of Social Influence in Agent Specialization, WASPS: A Weight Allocated Social Pressure System for the Emergence of Agent Specialization, A Genetic and Social Computational Model for the Emergence of Skill-Based Agent Specialization, A Social and Economic Model for Agent Specialization in the Simulation of Human Societies.

Paper [3] explores the idea of positive social influence leading to agent specialization where each agent possesses a genetic threshold for available tasks. When the thresholds for multiple tasks has been surpassed, agents will consider the choices of their neighbors in choosing which task to perform. Here, the probability of choosing a task would correlate with the number of neighbors performing that task. The more the neighbors, the higher the probability. It is shown that this approach leads to an increase in agent specialization when compared to the existing standard genetic threshold model. In [18], a social inhibition computational model is presented to show the emergence of specialization in agents. This model called the WASPS, allows agents to possess different skill levels for tasks and also allows agents to divide the resources among multiple tasks. The model uses the concept of social inhibition, which is a negative influence. Agents attempt to discourage neighbors from performing the same task. It is shown that the WASPS model increases the level of specialization in a random population of heterogeneous agents. The hybrid approach in [19] allows agents to respond to levels of stimuli in the environment. The hybrid model shows increased skill-based specialization allowing the emergence of specialization in populations of heterogeneous agents with different skills. In 'Social and Economic Model for Agent Specialization in the Simulation of Human Societies', the agents maintain the level of resource storage. The information regarding resource storage served as the primary factor for determining the agent's influence on its neighbors. With respect to economic sense, the increased specialization had a significant effect on the market and population.

2.2 Evolution of Agent Capabilities (Capability Evolution)

Agent Capabilities are task-directed/goal-directed subset of requirements, introduced to make the agent more adaptable/flexible to the environment. The capability concept has been studied and extended by researchers to develop its features such as its modularization [40]. In [21], the authors define capability as an identifiable unit which can be a set of plans, belief knowledge, associated rules and the recursive inclusion of other capabilities. In [23], it is discussed that an agent needs external capabilities because internal capabilities are not always enough for an agent to achieve its tasks/goals. The external agent capabilities are divided into two types. Firstly, those capabilities that can be achieved by collaboration with tools in the environment, and secondly, those capabilities that can be achieved with the association of other agents in the environment. A model that aims at extending capabilities of the agent with the help of other agents in the environment is proposed in [26]. Here, the social network is created using a weighted-graph (inclusion of distance cost between the nodes) method. And the agent aims to fill its capabilities by searching its network to find the required capabilities/required services with lower operational cost at minimum time. Also, a unique message sending mechanism has been proposed to propagate the agent's requirements to the other agents in the network through the agent's circle of friends. The best provider in the network is found using the domain knowledge and neighbor connectivity in the network. The results of their work prove that their algorithm can find near-optimal solutions in a very short time in comparison with the exhaustive search. The research is mainly aimed at extending the agent capabilities. In [43], the impact of social inhibition on the process of artifact selection is evaluated.

An artifact is a tool in the environment that has some capabilities and can be used by the agents to achieve their tasks. The authors simulated the effects of social inhibition and demands on the artifact selection by using a computational multi-agent model. The artifact selection in the presence and absence of inhibition are tested. As a result, the authors show the effect of social inhibition in capability selection. Also, the effect of demand on group performance is put forth. In [24], a model is proposed in which the agents exploit the desired artifact to achieve its goal. The knowledge about using the proper artifact of the agent is updated and the agent can improve its capabilities. Different types of evolutionary computation algorithms have been used for individual and social learning, that includes: Genetic Algorithm (GA), Cultural Algorithm (CA) and Multi-Population Cultural Algorithms (MPCA). These evolutionary computation algorithms have been used alongside with social network benefits and adaptation strategies to evolve capabilities in MABS. The author in [24] proposes a model in which the learning agent can learn to exploit proper artifact from the environment. An artifact can be seen as a set of external capabilities with special abilities that can help the agent to achieve its goals. Capability structure stores the knowledge about capabilities plans (select and use plans) of the artifact. The Performance Element selects the best artifact with respect to the plans in the capabilities and restriction in beliefs to achieve a goal. The belief space keeps the experience of using unsuccessful artifacts for each capability. The Learning Element is responsible for improving the capabilities with the help of its learning strategies. The intelligent agents requiring additional capabilities to perform a task are provided with extended capabilities using a reader module to the repository of capabilities (residing in the docking system of the network) [49]. [50] shows the cruciality of agent capabilities because of their autonomic self-management, self-adaptation, extensible knowledge, and flexibility. A MABS model that formulates capability evolution is proposed in [2]. The model emphasizes the evolutionary approach for capability extension/evolution. Evolutionary computation methods including GA, CA and Multi-Population CA supports the capability evolution at individual, population and multi-population levels. It is shown that the capability evolution is in regard to the evolution of society. Meaning, the significance of the capability evolution in societal evolution is stated in [2].

2.3 MABS and SNs

MABS can help experts to simulate and monitor the behavior of complex social systems and the inclusion of social networks in MABS models helps to solve a complex issue by breaking the issues into smaller pieces [8]. In recent years, this approach has been used to solve various types of problems in different fields such as marketing, anthropology, etc. A social system is a complex system comprised of a dynamic environment with varying constraints and growing/evolving population (individuals/agents). Individuals/Agents/Intelligent-Agents are designed to operate in such dynamic environments and mapping them to social networks proves to be helpful in making complex decisions at run-time. In a social network, nodes represent the agents, that is, the intelligent agents are mapped to the nodes of the social network and the relationship between the agents is depicted by the connections of the social network. Thus, enabling the agents to use the topological knowledge of the network accordingly to the problem at hand. In [51], the impact of network structure and its properties, such as maximum degree, average degree in addition with MABS properties in solving complex issues is discussed. A multi-agent based model to extend the agent's capabilities by using the advantages of the social network to minimize the operational costs and maximize the service quality is proposed in [1]. The model aims in filling its capabilities by searching its network to find its required services associated with lower operational cost. The external capabilities needed by agents is borrowed from service providers in the given network. An algorithm is proposed in [1], to assist the agent in the decision-making process by considering the agent's neighbor list as topological knowledge of the agent. It is shown that the inclusion of social networks results in an improved multi-agent framework. The results show that the algorithm can find the near-optimal solution/ ideal service provider in a very short time when in comparison with the exhaustive search. It is also shown that the performance of the proposed method with respect to the quality of services is much better when compared with the random selection method. In [52], the proposed model articulates the topological knowledge which is extracted from the structure of the network to form the normative knowledge (of the belief space of CA). The topological knowledge is used to direct and enhance the search process thus reducing the size of the required search space for finding the near-optimal solution. In [2], the proposed model is extended to address adaptation strategies for artifact use in unpredictable/uncertain environments with the aid of SNs. SN structures are introduced into the agent population and agents maintain static (or dynamic) social networks through which they communicate with other members of the population. Additional learning strategies of the agents are enabled through social networks. The model considers a multi-population setting where the agent migration is used to facilitate knowledge transfer between independently evolving social populations. Agents within each population learn both through social networks and their respective cultural belief spaces. The social learning strategies implemented here utilize a social network for agent communication towards learning. The social network may be an existing one or one that is constructed solely for learning artifact use. Social networks can remain fixed throughout the simulation or they can be dynamic in nature. The agents of this model do not commence learning with randomly generated actions, rather, these agents use the latest evaluated use action of their nearest neighbor as an influence to initiate the learning process with the aid of social network benefits.

2.4 MABS and Cultural Evolutionary Component

MABS model using social networks can be extended to use the evolutionary computation methods to develop learning and adaptation strategies for evolving capabilities. Researchers are constantly searching for unique ways to explain and provide insight into the complexities of human societal evolution [6]. A domain-independent computational evolutionary model for agent capabilities should prove beneficial to both new and existing MABS systems towards understanding the effects of domain knowledge and their exploitation in the evolution of complex societies.

Paper [2] emphasizes the strategies that take advantage of the social dimensions of MABS, where agents evolve through influence from other agents in the environment. Artifacts are reduced to a set of functional attributes whose values can be evolved by applying computational intelligence methods. Evolutionary approaches like genetic algorithms, cultural algorithms, and multi-population cultural algorithms aid in the evolution of artifacts in MABS. The proposed model supports artifact exploitation at individual, population and multi-population levels. The superiority of learned artifact use over random use is demonstrated in [63].

In papers [64][65] the evolutionary process in the context of optimization is discussed. Paper [66] shows the blend of evolutionary algorithms in multi-agent based modeling and the importance of evolutionary methods in decision support systems, mainly in dynamic environments is discussed.

2.5 Shortcomings/Limitations of the existing works

As we review the existing work in the field of computational modeling, it can be observed that none of the existing approaches model the synthesizing of ‘Agent Specialization’ and ‘Evolution of domain-independent agent capabilities in a dynamic environment by using the benefits of social networks and an evolutionary component’ techniques.

None of the existing works have used the concept of positive social influence exerted by experts in the network.

Very few research works in the field consider the dynamics of an environment (dynamics: changing external task stimuli, varying internal task thresholds, demand impact, etc).

Moreover, to the best of our knowledge, none of the existing works have blended the concept of capability evolution and agent specialization to improve the quality of individuals, in turn, is anticipated to improve the quality of the overall system.

With the widespread of social networks, it is possible to use its advantages to solve the open problems in the field much faster and accurately than the traditional approaches. Ultimately, the inclusion of the positive social influence factor exerted by experts in the network will help in creating specialized agents. We believe, exploring the system from a social network perspective and the inclusion of evolutionary component (CA) and positive social influence will result in an augmented/enhanced model where the agents can evolve into specialized agents, in turn improving the quality of agents(individuals) and quality of the overall system(population).

Chapter 3: Proposed Method

Our model to formulate the use of ‘positive social influence’ (PSI) exerted by influencers/experts/producer-agents in the network to design specialized agents and evolution of agent capabilities with the identification of best Producer-Agents using an evolutionary component (CA) is described here. The social system consisting of agents and an environment is seen as a network. The agent (Consumer and Producer-Agent) representation and their characterizations are described in detail in the next section. Then, the Consumer-Agents and Producer-Agents are mapped to the nodes of the social network and a detailed description of the network used is defined. Then, it is described how the components work with each other to evolve into specialized agents in an environment by exploiting positive social influence exerted by experts in the network. The cultural algorithm used to find suitable Producer-Agents in the network by using knowledge belief space is discussed in the latter part of this chapter.

3.1 Proposed Methodology

The proposed methodology builds a MABS model by integrating the benefits of social networks and cultural algorithms to aid the decision-making abilities of Consumer-Agents and the evolution of Consumer-Agent capabilities. The model exploits the ‘Positive Social Influence’ factor exerted by experts in the social network to mitigate ‘Agent Specialization’.

The social system under study has Consumer-Agents, Producer-Agents and a Task-set comprising of all the tasks within the system (Figure 1).

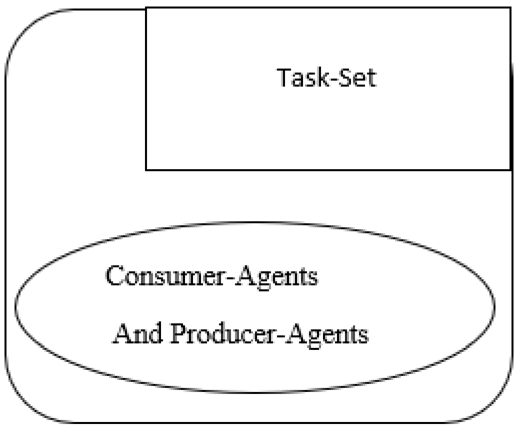

Figure 1 Social System Representation

In our multi-agent based simulation model, a social system with Consumer-Agents and Producer-Agents are mapped to social network of type small-world networks as they are more of a replica of the real-world networks and are presented using weighted graphs (Figure 2). A sample network with five Consumer-Agents and three Producer-Agents is presented here. The connecting edges have weights representing the distance cost between the nodes.

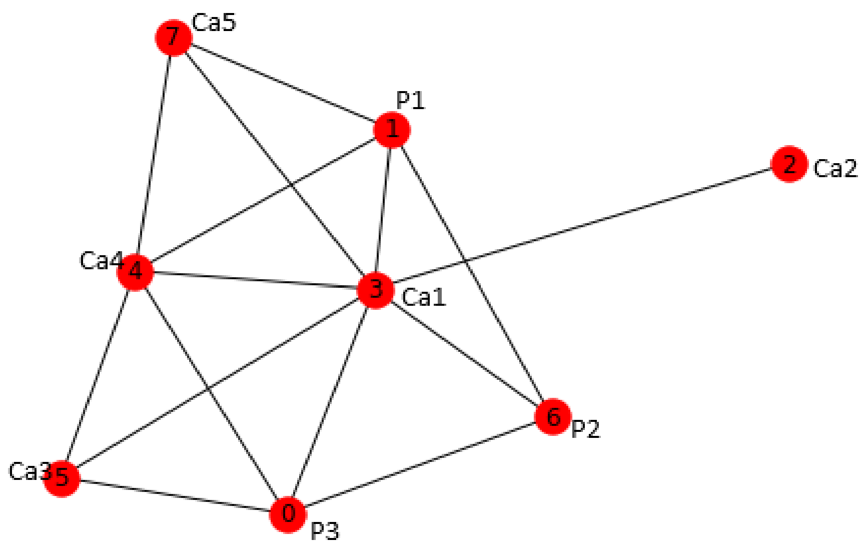

Figure 2 Consumer-Agents and Producer-Agents mapped to SN

The system with mapped agents (Consumer-Agents and Producer-Agents) and the task-set comprising all the tasks within the system/network is shown in Figure 3.

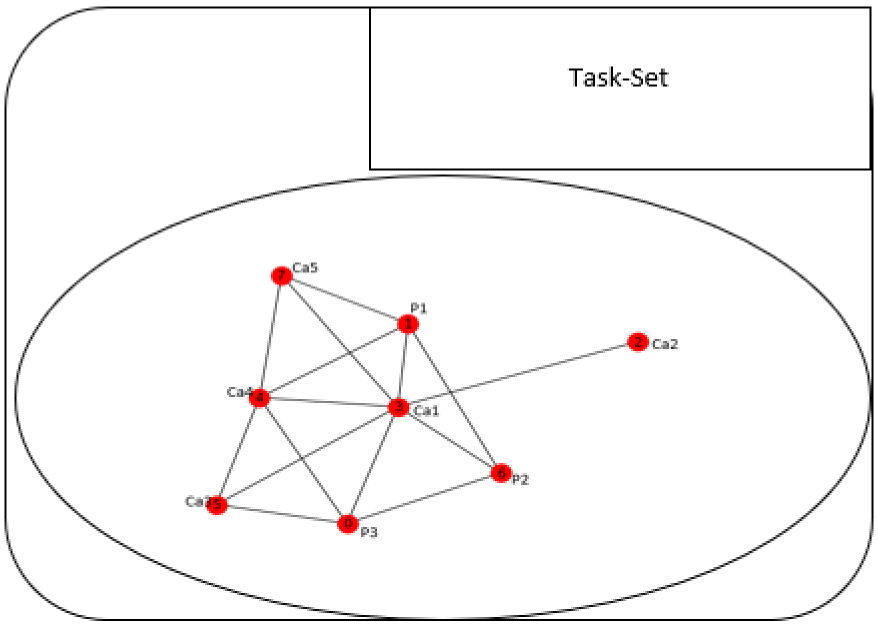

Figure 3 System with Task-set and mapped Agents

In social systems, examples varying from insect colonies with 20 to 40 specializations to the human environment where there are thousands of job choices [78]. That is, there are multiple tasks (Task-set) in a given social system that is to be selected and performed by the individuals in the population to evolve/grow over time. Initially, the Consumer-Agents needs to select a task from the task set available in the environment under various varying constraints. The social system under study is a replica of the human environment (increased number of job/specialization choices and growing population) and our simulation includes the concept of demand factor. Demand (∂) is the total amount of effort needed to satisfy all tasks in the system/environment relative to the total work ability of all the agents. In a system with low demand, the availability of agents to perform tasks is greater than current stimulus levels across all tasks [75]. That is, for demand $(\partial < 1)$, the stimulus across the tasks increases in smaller amounts when compared with the maximal amount of work the agents can perform. For simplicity, the demand was identical for all tasks. With the examples of demand parameters varying from $\partial = 0.7,\ \partial = 1\ and\ \partial = 1.3$, we explore the system under the demand parameter $\partial = 0.7$. This value for ∂ is selected as it is evident from the works [75] [18] [19], that the division of labor increased with this demand value for both increasing task number and increasing group size(agents). Division of Labor (DOL) can be described as a fundamental property in which the different class of agents within a system performs different tasks. The empirical evidence suggests that the division of labor in social systems/ social groups increases with increasing group size (agent population) [79] [80]. The measure of division of labor indicates the level of specialization. The DOL values less than 0.2 indicates the agents in the system are generalists, more than 0.6 indicates the emergence of specialization. The individuals(agents) performing the one task are called tasks specialist which is the higher level of specialization. The agents/social groups with DOL values (0.2 to 0.8) varied in the propensity of individuals to be task specialists or generalists [78]. We calculate the DOL values by using the method proposed in [74]. This method describes the procedure to calculate and quantify the degree to which the agents are specialized.

The specialization model used in our work is inspired from [3]. Our model is the extension of the basic concept adapted in the Response Threshold Model (Standard Genetic Threshold Model). Before describing the simulation mechanism implemented in our model, the brief of Response Threshold Model [75] is discussed for a basic understanding of the underlying model. In Response Threshold Model, the varying DOL values for the varying parameters of demand, stimulus intensity, task number and group size (agents) is calculated to understand which is the ideal condition that increases DOL values for the fixed response threshold model. A fixed response threshold indicates that the thresholds assigned to the individual agents remained constant (i.e., there is no self-reinforcement or forgetting). An internal threshold is assigned for all the available tasks in the system/environment for a given agent and is constant throughout the simulation run. For each simulation, the agents in the system select a task whose stimulus is higher than the corresponding intrinsic threshold. That is, the environment has tasks, each of which is associated with stimulus intensity and the agents has a threshold for all the tasks in the environment, once the simulation starts, the agents filter all the tasks whose external stimulus is higher than the intrinsic threshold and selects a task from the filtered set without any influence from the experts (the tasks are selected randomly). The Consumer-Agent's probability to stop performing a task is arbitrarily chosen to be 0.2. The stimulus of the tasks in the environment is updated using a formula [75] for every simulation run and the stimulus of a particular task is decreased by a constant value (chosen arbitrarily) once that task is selected by an agent in the system. The response threshold model explores the variability of response thresholds, group size, task number and demand value in influencing division of labor. The results of the response threshold model indicate that both low demand and high task number positively influences the emergence of specialization (increased DOL value) [75]. In [3], the concept of positive social influence is introduced where the system is seen as a network and the direct neighbors of the agents in the network influence the agents in selecting a particular task.

In our model, the concept of ‘Positive Social Influence’ is introduced in our social network, whereby the agent’s choice of specialization is influenced by the ‘Experts/Influencers/Producer-Agents’ in its neighbor list. ‘Positive Social Influence’ is a concept where an agent’s choice of specialization is influenced by the choices of those within its social network. With an option of multiple specializations available in a given system, we factor in what the agent’s expert neighbors are doing and allow that to influence the Consumer-Agent’s decision. Study shows that in most of the cases, the introduction of ‘Positive Social Influence’ in a network causes an increase in the level of agent specialization in comparison with the system with no social influence. Specialization allows individual agents to maximize their productivity. The example of tasks in a given environment includes, a student becoming a ‘Computer Engineer’, an individual becoming a ‘Sports Player’, etc. A Consumer-Agent selects the task that is influenced by the agent’s network expert neighbors. Before each simulation run, every Consumer-Agent is given a list of all the tasks in the environment and each task is associated with a threshold θ which is arbitrarily chosen as θ = 50 with standard deviation ranging from (-5 to +5). The standard deviation can vary from 0 to 15, but here each agent is associated with a threshold whose deviation ranges from -5 to +5, as the amount of threshold level variation between the agents is irrelevant (as long as some variation exists). For simplification, the threshold value and deviation are assumed to be identical. This does not mean that the thresholds for all available tasks for a given Consumer-Agents were identical. That is, the thresholds assigned to given Consumer-Agent remained constant for the whole simulation run. Each task in the environment is associated with a stimulus. ‘S_t’ is the amount of stimulus available with the task ‘t’. First, the Consumer-Agent filters all the possible tasks in set T, where the stimulus of the task in the environment is greater than the threshold associated to the task in its task list. Then the Consumer-Agent selects a task using two methods compared in this work: Standard Genetic Threshold Model and Positive Social Influence Model. In the standard genetic threshold model, the consumer-agent selects a task without any influence from the experts (randomly) from the filtered set T. In our positive social influence model, the task is selected from T, which is influenced by the agent's expert neighbors.

Given the set T, Consumer-Agent selects one task t ∈ T having probability:

$$\frac{(1+\psi Nt)}{\sum(\psi N+\# T)}$$

for all tasks t in T. The formula is similar to the one used [3]. Here, for a set of tasks T, Nt represents the number of agent's expert neighbors currently engaged in task t, N represents the number of active neighbors and #T holds the value of total number of tasks available in the environment. Symbol ψ (psi) represents the influence impact an agent has on its neighbour selection. The value of ψ is arbitrarily set to 0.5, as the tests conducted in [3] shows no pattern from varying this value. The concept of demand ∂ is included in our simulation. Demand is the total amount of effort needed to satisfy all the tasks relative to the total work ability of all the consumer agents in the environment. Here, the demand level of 0.7 is tested as the results from the previous works [75] [18] [19] shows that the DOL values are improved with low demand values for higher task numbers. We are interested in understanding the influence of factors (like demand, stimulus) on a system with higher task numbers and increased agents as we are considering the human societal environment where the environment has thousands of job choices (tasks) and increasing number of individuals.

At the beginning of each iteration, the stimulus level of each task is updated. Initially the stimulus value is set to 0. For every iteration, the stimulus value is updated(increased) using the formula

$\alpha \left(\frac{N}{T}\right) \partial$, where α is arbitrarily chosen and assigned as 3, N is the number of Consumer-Agents, T is the number of tasks and ∂ is the demand level. Once a task is selected by the Consumer-Agent, the stimulus of that task in the environment is decreased by the value of α (i.e. 3). Meaning, the stimulus level for a task is reduced when a Consumer-Agent selects that task. Therefore, it is possible to exhaust the demand for each task, especially when the demand level is below 1. Once a task is selected, the individual agent can spontaneously cease working on that task with a probability of $\left(\frac{1}{\tau}\right)$.

The value of τ is 5 (time steps, arbitrarily chosen). That is, an individual agent performs any given task for an average duration of five time-steps. Meaning, the probability of an agent to stop performing a task is arbitrarily chosen to be $\left(\frac{1}{\tau}\right)$, that is, 0.2. The frequency of the selected tasks is used to determine the division of labor values, which in turn determines the level of specialization of the agents. Our system with Task-set associated with external stimulus and their respective requirement capabilities, Consumer-Agents associated with an internal threshold for tasks, and Producer-Agents with influencing expertise and serving capabilities are shown in Figure 4.

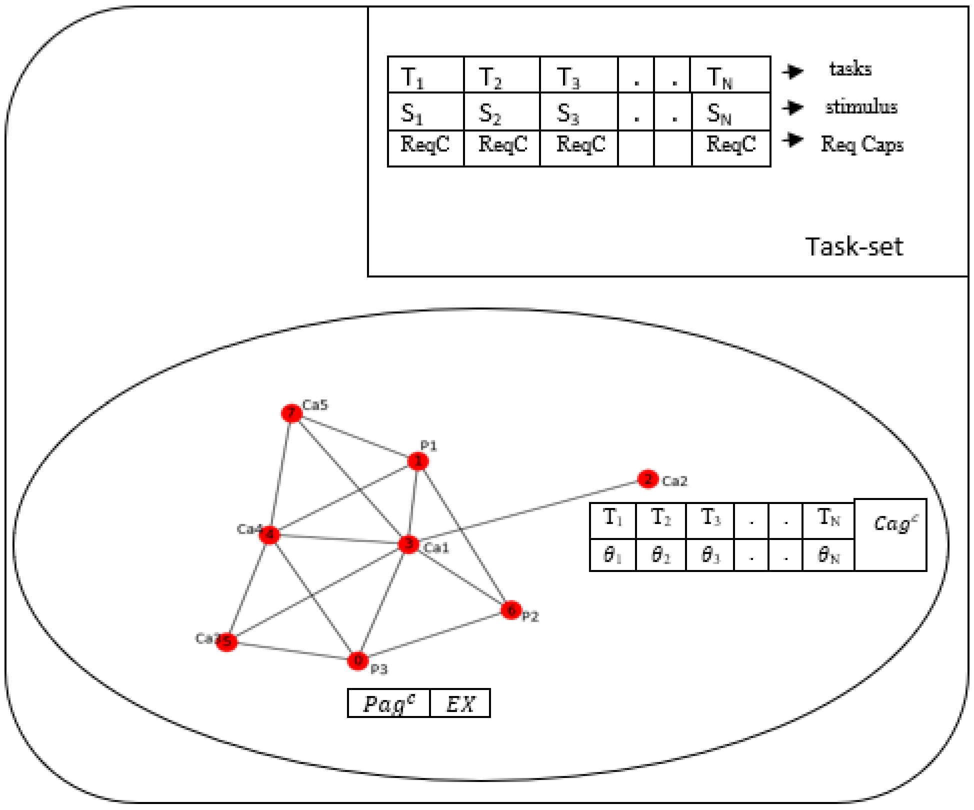

Figure 4 System with Task-set, Consumer-Agent and Producer-Agent Representation

The Task-set shows the associated stimulus with every task within the system and the 'Req Caps' (Required Capabilities) to achieve a particular task. The Producer-Agent services capabilities and has an EX (expertise) influencing the direct neighbors. The Consumer-Agents have respective capability set and the value of the internal thresholds (θ) associated with each task (Figure 4). As the Consumer-Agents decisions are based on probability, there still remains a chance for the Consumer-Agent to choose a specialization that none of its expert neighbors have chosen. Also, when the agent's neighbors are consistent and are maintaining their specialization, then there is a higher probability of it choosing the popular tasks determined by expert neighbors. A detailed explanation of the embraced methodology can be found in [3] [75].

In this work, the expert/influencer/Producer-Agent neighbors are considered rather than considering all the network neighbors to influence the agent's decision. In [3], the influence of the agents is impacted by all its neighbors. Ideally, all the neighbors in the network may not be experts or influencers. Hence, the impact of influencers alone is considered in this model. The DOL measure of the recorded frequency of tasks for all the agents for every simulation run is calculated using the method proposed in [74].

The active Consumer-Agents record their specialization (their selected tasks) at the end of each iteration. The frequency of the tasks selected by the consumer-agents recorded is then stored in a n x m matrix, where n indicates each Consumer-Agent and m indicates each task. The matrix is normalized such that the sum of all the cells in the matrix is 1. The DOL value is calculated by dividing the mutual entropy score with Shannon's index of the matrix. Each cell of the matrix contains the frequency with which a specific Consumer-Agent was observed performing a specific task, and the matrix is normalized. The Shannon index, Htasks for the distribution of individuals across tasks [74] and the mutual entropy [81] of the entire matrix are calculated. Dividing mutual entropy with Shannon's index Htasks yields the resulting index.

The result (DOL value) indicates how specialized consumer-agents were, ranging between 0 and 1. A score of 1 indicates that the agents are fully specialized, and zero indicating no specialization at all.

The evolution of capabilities is addressed with the identification of suitable Producer-Agents in the network using an evolutionary algorithm (CA). Realizing the best decision-making process that improves the degree of specialization for different number of tasks, varying group size (agents) and a varying number of influencers in the network, an ideal environment with varying parameters (number of tasks, number of experts, number of Consumer-Agents) is passed as input to our CA. Though the selection of tasks/specialization of consumer-agents is influenced by experts/influencer/producer-agents in the network, it is not necessary that the consumer-agents has to acquire the required capabilities to achieve its tasks from the influencing experts/influencers.

Also, the influencing Producer-Agent in the network is not necessarily the Producer-Agent providing the required capabilities. For example, let’s consider a student Consumer-Agent and a parent influencer (Producer-Agent). The student is influenced to select Computer Engineering as a specialization from the influence of parent (Computer Engineer Professional). Few capabilities of acquiring this selected specialization include coding, problem-solving ability, technical skills, and research skills. Now, the student has to acquire the required skills from suitable Producer-Agents in the network. The suitable producer-agents in the network include teachers, professors, coaching centers, universities, etc. With the aid of an evolutionary algorithm, we can find suitable Producer-Agents in the network. A Producer-Agent can provide a single capability service or multiple capability services; therefore, more than one Producer-Agent might be needed to support every consumer-agent. Suitable Producer-Agents are a team of Producer-Agents/ a Producer-Agent providing capability services at minimum operational cost and minimum distance cost.

Once the task is selected by a Consumer-Agent by considering ‘Positive Social Influence’ factor exerted by the experts, the required capabilities to achieve its selected task are listed. As the Consumer-Agents has minimal capabilities at initialization, more so often, they do not possess all the required capabilities. Evolution of Consumer-Agent capabilities can be approached as a problem to acquire the missing capabilities from suitable Producer-Agents to attain their selected tasks. As discussed, every Producer-Agent has a predefined set of capabilities to serve and each capability is associated with an operational cost. The influencers can also be the Producer-Agents providing services or not in a given network.

In this problem of identifying a team of Producer-Agents or a Producer-Agent in a social network, each Producer-Agent possess a set of capabilities (or a capability) to offer/service. As Consumer-Agents selects tasks (this task information is collected from the matrix from the specialization model), the required capabilities are to be acquired to achieve its selected tasks and the suitable Producer-Agents servicing the required capabilities are identified using an evolutionary CA. The success of task achievement depends on finding a team of suitable Producer-Agents that services all the required capabilities for the selected task. Our system with Task-set representing requirement capabilities, Consumer-Agent with capability set and Producer-Agents with servicing capabilities are as shown in Figure 5.

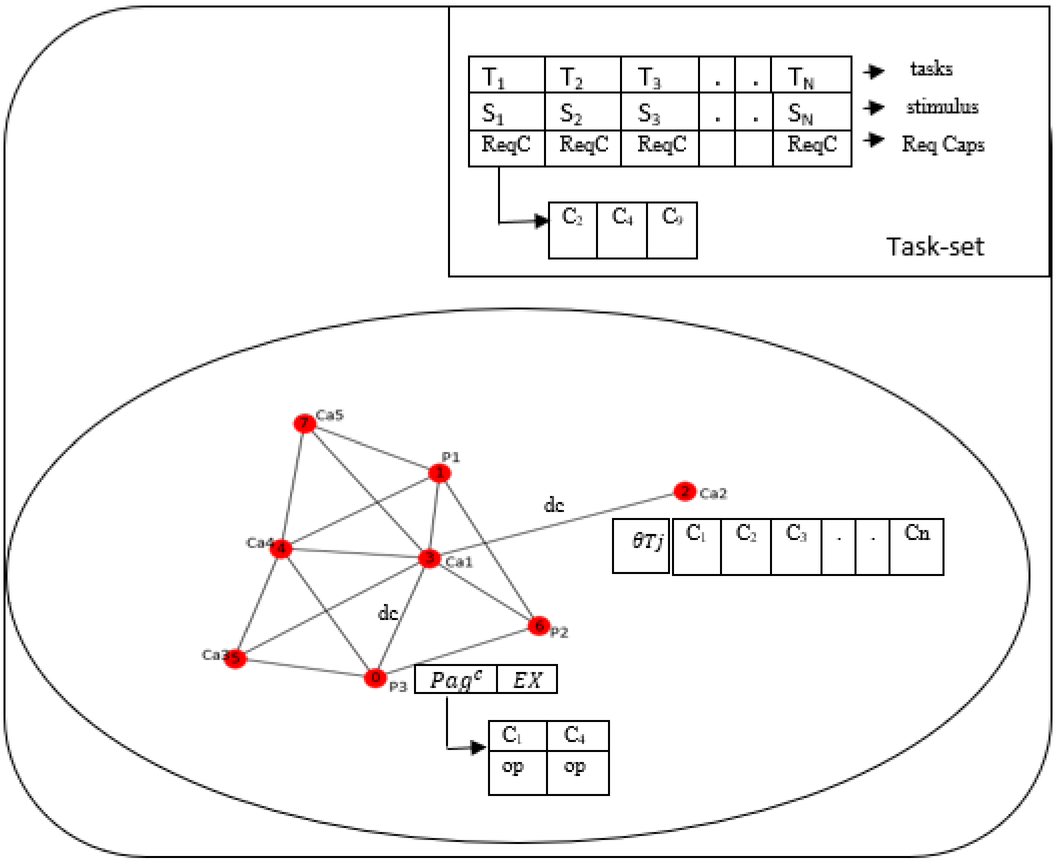

Figure 5 System with Task-set, Consumer-Agent and Producer-Agent Capability Representation

Task-set comprises of the tasks, associated stimulus and respective requirement capabilities, Consumer-Agents has a capability and threshold associated with tasks and Producer-Agents has EX (expertise) information and servicing capabilities with associated operational cost. The 'dc' associated with the edges represent the distance cost (Figure 5). The Consumer-Agent selects a task, the required capabilities to achieve the selected task is predefined. The Consumer-Agent possess a set of capabilities initially, the difference of the required capabilities and the possessed capabilities are the missing capabilities that are to be acquired. The missing capability set is represented using a list and if three capabilities are missing, then the length of the list is 3. For each missing capability, the value of the cell of the list is selected from a set of Producer-Agents which services/provide the capability.

CA aids in identifying suitable Producer-Agents using adaptation strategies and by exploiting the benefits of social networks. CA consists of a population space and a belief space with a communication protocol between them. The suitable team of Producer-Agents (producer-agent) is identified by executing a series of iterations. The population space is used to maintain the current list of generated teams of Producer-Agents in each iteration. At the start state of the algorithm, the predefined set of random Producer-Agent teams is generated to form the initial population. A team (individual) in the population space represents a candidate solution to the problem. Now, for each cell of missing capability, we have a set of producer-agents who provide services to that particular missing capability. The suitable Producer-Agents are the ones which provide services with minimum operational cost and minimum distance cost. Here, the distance between the Consumer-Agent and the Producer-Agent in the network is considered as the distance cost. Consumer-Agents and Producer-Agents mapped to a weighted graph and is given as input to the algorithm (CA). The weights represent the distance cost. The shortest weights are precomputed. The fitness function assigns the fitness score to the candidate solutions in the population space by considering operational cost and distance costs. The fitness score is maximum for the minimum operational cost and minimum distance cost values. The fitness score determines how fit the solution is. From the considered initial population in the population space, the new population is generated by applying evolutionary algorithm operations like selection, crossover, and mutation on the first iteration. The top teams selected constitutes the knowledge of belief space. In the next iteration, the selected population space and the knowledge from belief space influence the new population creation and so on. At the end of a given number of iterations, the top team from the candidate solutions is selected and is the near-optimal suitable Producer-Agent team for the respective missing capability list of a Consumer-Agent.

That is, the weighted graph of Consumer-Agents and Producer-Agents of the network is given as input. Each Consumer-Agent has information about the selected task and has a set of capabilities. The Producer-Agents in the network possess an expertise and offer the capabilities with respect to the expertise. Every task has a predefined set of required capabilities. The missing capability of each Consumer-Agent with respect to the required capabilities is listed. For every missing capability, the Producer-Agents servicing that particular missing capability is listed. A single Producer-Agent can service/provide more than one capability. For a list of missing capabilities, each cell of missing capabilities has a list of Producer-Agents servicing the capability. The combination of the Producer-Agents for the given list of missing capabilities constitute the candidate solutions of the initial population space. For every candidate solution, a fitness score is assigned using a fitness function, where the function assigns minimum fitness score for the Producer-Agents with minimum operational cost and minimum distance cost. The teams (candidate solutions) are sorted in ascending order with respect to the fitness score, such that the solution with minimum fitness score is at the beginning of the sorted list. A percentage value of the best-fit solutions from the sorted list of candidate solutions is selected (algorithm operation: Selection) and is subjected to crossover and mutation. In the next iteration, the same procedure is repeated, and a percentage (or a predefined number) of best top team solutions of the sorted list generated constitutes the knowledge belief space. From the next iteration (generation) onwards, the new population is generated from the previous ‘selected population (selection)’, ‘applying crossover and mutation’ and ‘knowledge from belief space’. The same process is repeated for the given number of iterations (generations). The knowledge obtained from the belief space guides the search direction and aids in evolving the team of Producer-Agents faster when in comparison with exhaustive search, random search, and basic genetic algorithm (GA). At the end of generations, the top team is selected as a suitable team of producer-agents from which the consumer-agents can acquire the missing capabilities to achieve its selected task. The agent representation and characterization, the sample social network mapping of consumer-agents and producer-agents, and the algorithm proposed to identify the suitable producer-agents are represented in the forthcoming sections.

3.2 Agent Representation and Characterization

Our social system comprises of Consumer-Agents, Producer-Agents and a set of tasks global to the network environment. The representation and characterization of Consumer-Agents, Producer-Agents and Task-set are presented here.

3.2.1 Task-set

The tasks in the environment are associated with stimulus levels whose value is subjective to the demand factor in the system/network. Demand represents the total effort required to complete all tasks relative to the available total effort from the Consumer-Agents. In a system with low demand, the availability of Consumer-Agents to perform the tasks is greater than the current stimulus levels across all tasks. Thus, the demand of the system decreases with the increase in the group size (increase in the number of Consumer-Agents). For $\partial < 1$, the stimulus across the tasks increases less than the maximal amount of work the system/environment/Consumer-Agents can perform. Since we consider the human environment as our social system, there are more tasks and Consumer-Agents in the system and hence low demand levels are considered. We consider a demand level of $\partial = 0.7$, and the stimulus in the environment is zero initially and updated with stimulus updated formula for each simulation cycle. The stimulus update formula is $\alpha \left(\frac{N}{T}\right) \partial$, where α is arbitrarily chosen and assigned as 3, N is the number of Consumer-Agents, T is the number of tasks and ∂ is the demand level. Once a task is selected by the Consumer-Agent, the stimulus of that task in the environment is decreased by the value of α (i.e. 3). Meaning, the stimulus level for a task is reduced when a Consumer-Agent selects that task.

Task-set is the collective representation of tasks available to perform in a given environment and each task is defined as $t \in \mathrm{TS} \triangleq (\mathrm{tid}, S, \mathrm{ReqC})$ where TS denotes the Task-set, tid denotes the task id, S denotes the stimulus intensity and ReqC represents the required capabilities to perform the task and is defined as $ReqC = (reqc1, reqc2, reqc3 \ldots . reqcn)$ where reqc1, reqc2, reqc3 till reqcn represents the abilities required by a Consumer-Agent to achieve the selected task.

3.2.2 Consumer-Agents

Consumer-Agents are intelligent agents commencing their role as consumers in the system. Consumer-Agents (individual, student, child, sports player, etc) selects the task by considering the external stimuli associated with the tasks in the environment, its internal thresholds associated with the tasks of and the positive social influence exerted by the experts. Also, Consumer-Agents consumes the services(capabilities) from the Producer-Agents to evolve over time. Every Consumer-Agent is characterized by an initial set of capabilities, a task set with associated thresholds and is defined as,

$$ag^c \in C_{AG} \triangleq (Cag^c, \mathrm{T}_T)$$

where C_{AG} represents Consumer-Agent class, Cag^c represents the capability set of the Consumer-Agent and is defined as $Cag^c = \{c1|c2 \ldots . |cn\}$ where c1, c2 till cn represents the abilities possessed by the Consumer-Agents and T_T denotes the tasks and the corresponding threshold available for each task. That is, for every task in the environment, a threshold is assigned. T_T is defined as, $T_T \in (\theta T_j)$ where θT_j is the threshold associated with task T_j and the value j represents the task id of every task in the environment.

3.2.3 Producer-Agents

Producer-Agents are providers of services/capabilities in the system. Examples of Producer-Agents in a network can include a teacher, parent, sports coach, an institution, etc. Each Producer-Agent is defined as,

$$ag^p \in P_{AG} \triangleq (Pag^c, EX)$$

where P_{AG} represents Producer-Agent class and Pag^c represents the capability set offered by the Producer-Agent and is defined as,

$$Pag^c = \{c9, op|c3, op \dots . |cn, op\}$$

where c9, c3 till cn represents the capabilities/services offered by the Producer-Agents and ‘op’ associated with every capability/service denotes the operational cost/service charge of the capability provided/serviced. *EX* represents the expertise of the Producer-Agent. The associated operational cost can vary from 1 to 10 (1 <= *operational cost* <= 10).

3.3 Mapping of a Social System to a Social Network (Network Representation)

A social network defines connectivity between individuals in a population. The agents (Consumer-Agents and Producer-Agents) in our social system are mapped to the nodes of the social network. The network is defined as,

$$N \triangleq ag^c \cup ag^p$$

Where ag^c represents the set of Consumer-Agents and ag^p represents the set of Producer-Agents respectively. Each agent in the network is linked to one or more agents forming a graph.

Though the network/graph can be created randomly, the likelihood of hub nodes created by random approach results in a poor reflection of real-world social networks. There are multiple ways of creating a small-world network that replicates real-world social networks. Note that the small-world networks should also be scale-free networks whose degree distribution follows a power pattern, as is very common in real-world networks.

As the unweighted graph does not show the level of connection between the agents, hence a weighted graph is considered. The geographical distance between the nodes of the network constitutes the weight of the edges between the nodes. Therefore, the weight here can be interpreted as a geographical distance cost between the agents. The weight/distance cost is the value between 1 to 10 (1 =< *distance cost* <= 10) and can be interpreted as the geographical closeness of agents in the network.

3.3.1 Algorithm: Pseudocode to find Suitable Producer-Agents

Input: Graph G (Consumer-Agents and Producer-Agents mapped as network), and Task-set.

Output: Suitable Producer-Agent team who provide services with minimum operational and distance cost

```
1. gen ← number of generations, pop_size ← size of population, tp ← number of top
   performing teams for belief space knowledge, et ← number of elite teams for next
   generation, population ← Initialize the population with candidate solutions
2. Begin
     i.  for i ← 1 to gen do
    ii.          calculate fitness score for candidate solution in population
                 {
```

```
                fitness_function(population)
                for j ← 1 to pop_size do
                    #Applying fitness_score
                    population(j)_fit = sum (operational cost + distance cost)
                end for
                }
      iii.      sort population based on fitness score
      iv.       for k ← 1 to tp do
                    BeliefSpace ← (population(k))
                end for
      v.        Select elite team of candidate solutions from BeliefSpace
                    BeliefSpace(et)
      vi.       Apply Selection
                    selection(population)
      vii.      Apply Crossover
                    crossover(population)
                        parent_1 ← random(population)
                        parent_2 ← random(population)
                        # select winner and loser with respect to fitness_score
                        child ← crossover of loser parent with winner parent
      viii.     Apply Mutation
                    mutation(population)
                        # Apply mutation on randomly selected candidate solution
                        mutated_child ← mutation(random(population))
      ix.       Create new population
                    new_population = add (selected_population, BeliefSpace(et), child,mutated_child)
                    population ← new_population
      x.    end for
  3. suitable_producer_agents ← population[0]
  4. End
```

There are multiple Consumer-Agents and Producer-Agents in the network graph. For every task selected by the Consumer-Agent from the Task-set, the required capabilities are listed. The

Producer-Agents servicing each capability is listed. The combination of Producer-Agents (candidate solutions) for the required capability set constitutes the population space of our algorithm. The suitable Producer-Agent team is selected through iterations/generations using evolutionary operations like selection, crossover, and mutation. For each iteration (gen), the population of a given size (pop_size) is updated as the population and the fitness score for each candidate solution in the population is assigned using a fitness function. The fitness function calculates the sum of operational cost associated with the producer-agents in the candidate solution and distance costs between the Consumer-Agent and all the Producer-Agents in the candidate solution. The lesser the fitness score, the fitter the solution. The population is sorted in ascending order, such that the first candidate solution in the population is the one offering services at minimum operational and minimum distance cost, the second solution is the next best and so on. Note that the evolutionary algorithm provides the near-optimal solutions and hence the minimum operational cost here does not mean the least possible operational cost. From the sorted population, we select the top teams of a specific percentage value, called the selection operation. The top teams of a given size (tp) is updated to the Belief Space knowledge. The population of size et (number of elite teams) is created using the top team performers in the Belief Space. Then the evolutionary operations, crossover, and mutation are applied to the population. In crossover, the parents are selected randomly from the population. Each parent is considered a winner or a loser with respect to the fitness score. If the fitness score of the parent_1 candidate solution is lesser than the fitness score of the other parent, then parent_1 is the winner and parent_2 is the loser. We apply crossover only to the loser parent. A part of the loser parent's candidate solution is swapped with a part of the winner parent to create a child. If the created child's fitness score is less than the loser parent, then the child is added to the population for the next generation. In mutation, a random candidate solution from the population is selected and a part of that solution is replaced with a part of another solution from the population (mutated) to create a mutated child. Mutated child is a solution created with a random mutation. Now, the population from the selection process, belief space, crossover (child) and mutation (mutated child) is gathered to create a new population for the next iteration. The same process of evolutionary operations is implemented over a given number of generations and the best Producer-Agent team from the population is selected as a suitable solution (suitable Producer-Agent team).

3.3.2 Identifying Suitable Producer-Agents

In a network mapped with Consumer-Agents and Producer-Agents, identifying the suitable Producer-Agents for acquiring the required capabilities needed to achieve the Consumer-Agent selected task is discussed in detail here.

The network considered is a weighted undirected graph with weights depicting the distance cost between the nodes. Consumer-Agents and Producer-Agents are mapped to the nodes of the graph. There are 'C' number of Consumer-Agents and 'P' Producer-Agents in the network and is represented as, ConsumerAgents = $\{ca_1,ca_2,\ldots ca_c\}$ and ProducerAgents = $\{p_1,p_2,\ldots p_p\}$ respectively. Each Consumer-Agent selects the task from a set of tasks in the network under the influence of positive social influence exerted by the influencer/experts/producer-agents. Task-set is the collective representation of tasks available to perform in a given environment and each task is defined as $t \in \mathrm{TS} \triangleq (\mathrm{tid}, \mathrm{S}, \mathrm{ReqC})$ where TS denotes the Task-set, tid denotes the task id, S denotes the stimulus intensity and ReqC represents the required capabilities to perform the task and is defined as, $ReqC = (reqc1, reqc2, reqc3 \ldots . reqcn)$ where reqc1, reqc2, reqc3 till reqcn represents the abilities required by a Consumer-Agent to achieve the selected task. Once the task from the Task-set is selected by the Consumer-Agent, the required capabilities (ReqC) to achieve the task is listed. The Consumer-Agent has a minimal set of capabilities initially, $\mathrm{Cag}^{\mathrm{c}}$ represents the capability set of the Consumer-Agent and is defined as, $Cag^c = \{c1|c2 \ldots . |cn\}$ where c1, c2 till cn represents the abilities possessed by the Consumer-Agents. The required capabilities which are not possessed by the Consumer-Agents are to be acquired from the suitable Producer-Agents in the network. That is, the required capabilities needed to be acquired are listed (ReqC – Cag^c).

For example, if a Consumer-Agent ca_1 selects a task t_1 then the required capabilities to achieve t_1 is defined in ReqC of t_1 and the difference in ReqC of t_1 and Cag^c of ca_1 is listed. Let Cag^c of ca1 have a capability set of {c1,c3,c5,c6,c7} and ReqC = {c1,c2,c3,c4,c8}, then the difference in the sets gives the list that needs to be acquired. Therefore, (ReqC – Cag^c) gives the needed required capabilities {c2,c4,c8}.

From ProducerAgents, the suitable Producer-Agents servicing the needed requirement capability set is identified using an evolutionary algorithm (CA). For every needed requirement capability, all the Producer-Agents servicing the capability is listed. Then the producer-agents are assigned to every capability. There can exist many such combinations of Producer-Agents, however, we are interested in the team of Producer-Agents with minimum operational cost and minimum distance cost. A ProducerAgent can have one or multiple capabilities to service and can support multiple ConsumerAgents. For each capability required there are multiple ProducerAgents in the network to service the capability. Example, capability c2 is serviced by {p4, p2, p8, p9, p12}, capability c4 is serviced by {p3, p4, p8} and c8 is serviced by {p1, p2, p6, p7, p11, p16}. Pictorial representation of the same is as shown.

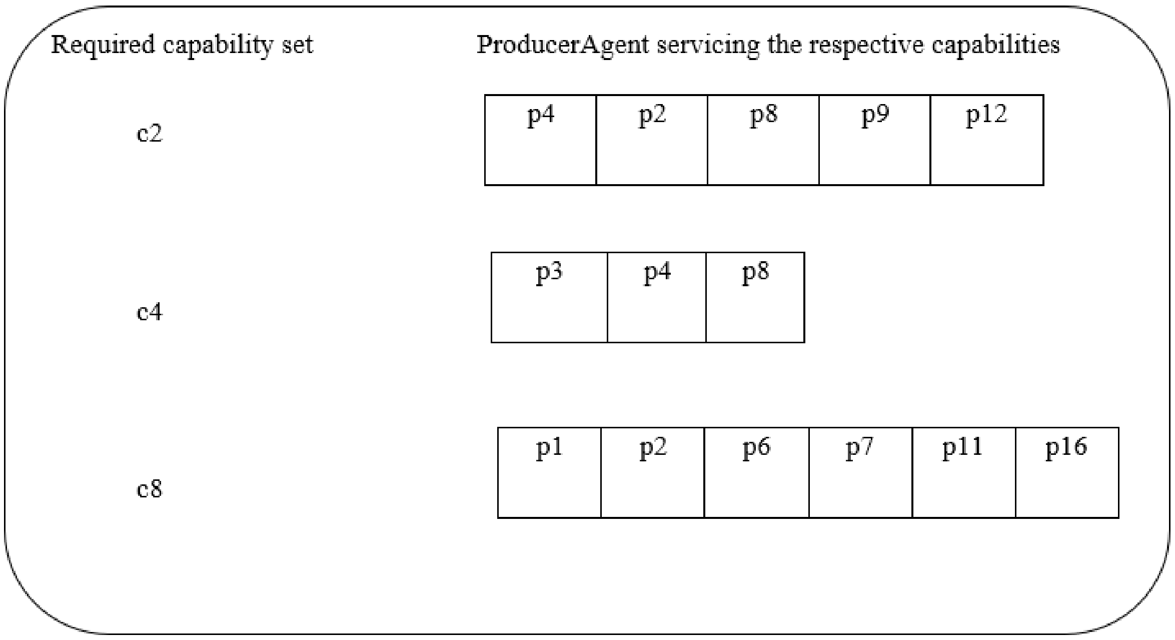

Figure 6 Example Producer Agent servicing required capabilities

The combination of Producer-Agents for required capability set constitutes the population space of our algorithm and are the candidate solutions to our problem of identifying the suitable Producer-Agents for the required capability set. That is, the combination of Producer-Agents for the capability set {c2,c4,c8} includes {(p4,p3,p1), (p4,p3,p2), (p4,p3,p6), (p4,p3,p7) ...(p12,p8,p16)}. For a capability set of length three, there are 90 (5*3*6) possible solutions from the given combination of Producer-Agents. The best suitable team of Producer-Agents from the combination of ProducerAgents is to be selected by considering minimum operational cost and minimum distance cost. Operational cost is the cost associated with every capability serviced by the producer-agent ranging from 1 to 10 (1 <= *operational cost* <= 10). Whereas, distance cost is the edge weight of the edge connecting the consumer-agent node and the producer-agent node ranging from 1 to 10 (1 =< *distance cost* <= 10). Here the shortest path values between all pairs of nodes are precomputed. For a given combination of a team of Producer-Agents, the collective operational cost and collective distance cost between the Consumer-Agent node to the Producer-

Agent nodes is calculated. The sum of operational cost and distance cost constitutes the fitness score. The lesser the fitness score, the fitter the solution.

The population space of the proposed CA is populated with the candidate solutions. The fitness score of each candidate solution is assigned by the fitness function which calculates the operational cost and distance costs. The fitness score is the sum of operational cost and distance cost. The solutions with minimum fitness score are sorted, and the top teams are selected to constitute the knowledge of the belief space. Belief space has the top teams of candidate solutions and guides the search direction and evolves the teams faster than the basic genetic algorithm. The combination of candidate solutions from the top teams from Belief Space is created, and the elite teams of specified number from the belief space is added to the population. For a given number of generations, the process of finding the suitable Producer-Agent team with minimum operational cost and minimum distance cost using evolutionary operations like selection, crossover, and mutation is repeated. For every iteration/generation, the new population is generated from the previous ‘selected population (selection)’, ‘applying crossover and mutation’ and ‘knowledge from belief space’. Crossover is implemented by selecting two random parents from the solution. The one parent with a minimum fitness score is considered the winner and the one with lower fitness score is considered the loser. Part of a loser parent is swapped with part of a winner parent to create a child. The fitness of the created child is calculated and is added to the population if the fitness score of the child is lesser than the fitness score of the loser parent. Mutation is implemented by selecting a random candidate solution and applying mutation, that is, a part of the selected candidate solution is replaced from another random candidate solution from the population. At the end of generations, the best candidate solution with minimum operational cost and minimum distance cost is selected.

Chapter 4: Implementation and Experimental Setup

A MABS model to simulate a social system consisting of Consumer-Agents and Producer-Agents and Task-set in the environment is implemented using ‘Mesa Agent-Based Modeling Framework’ (Python) [76] to show the emergence of agent specialization. To imitate a network for our social system, dynamic social networks using the Barabasi-Albert [77] model is created.

All the real-world networks thought to be scale-free networks. Scale-free networks are a type of network characterized by the presence of large hubs and are the ones whose degree of distribution follows a power law. In such power-law distributed networks, the nodes of the network are not evenly connected. As the creation of random networks ends up being a poor reflection of real-world networks, we use the Barabasi-Albert model. Barabasi-Albert model creates scale-free networks, wherein all the agents are connected to all other agents, but not necessarily directly.

Once our social system is mapped using a social network structure, the preliminary effort is taken in task delegation/selection. The system is subjected to the demand factor, where the value of demand more than 1 $(\partial > 1)$represents that there is more work than the collective efforts of all the agents (consumer-agents), the value of demand equal to 1 $(\partial = 1)$ means that the work required and effort needed to satisfy the work is equal and the value of demand less than 1 $(\partial < 1)$ indicates that there is less work available than can be performed by all the agents. The demand value of 0.7 is selected in our simulation as it is proven to increase the emergence of agent specialization for a social system where the number of tasks and the number of agents is relatively more. We consider the human environment, where there are more tasks/jobs in the environment and the group size/population/agents in the human environment are growing, hence the demand level of $\partial = 0.7$ is considered in our simulations. Every task from the Task-set in the environment is associated with stimulus intensity. The stimulus of each task is updated using stimulus update formula $\alpha\left(\frac{N}{T}\right)\partial$, where α is arbitrarily chosen and assigned as 3, N is the number of Consumer-Agents, T is the number of tasks and ∂ is the demand level. Once a task is selected by the Consumer-Agent, the stimulus of that task in the environment is decreased by the value of α (i.e. 3). Meaning, the stimulus level for a task is reduced when a Consumer-Agent selects/consumes that task. Initially, the stimulus levels of all the tasks are set to zero, in each iteration the value of the stimulus is updated and decreased accordingly. Every Consumer-Agent is associated with an internal threshold for every task in Task-set. That is, every Consumer-Agent is associated with a threshold θ which is arbitrarily chosen as θ = 50 with a standard deviation ranging from (-5 to +5) for every task.

At the beginning of the simulation, every Consumer-Agent filters all possible tasks in the environment which meets the agent threshold. That is, the tasks whose stimulus level is more than the threshold value of the task in Consumer-Agent is filtered, and the task from this filtered set is selected using two methods, standard genetic threshold method, and positive social influence method. In the standard genetic threshold method, a task is selected from the filtered set without any influence from the experts (a random task from the filtered set is selected) and in our method the task is selected using positive social influence exerted by the neighboring experts/influencer/Producer-Agents. The formula to select a task using positive social influence is, probability of

$$\frac{(1+\psi Nt)}{\sum(\psi N+\#T)}$$

Here, for a set of tasks T, Nt represents the number of agent's expert neighbors currently engaged in task t, N represents the number of active neighbors and #T holds the value of total number of tasks available in the environment. Symbol ψ (psi) represents the influence impact an agent has on its neighbor's selection. The value of ψ is arbitrarily set to 0.5. At each iteration, an agent has a chance of $\left(\frac{1}{\tau}\right)$ to switch specialization. (The value of τ is arbitrarily set to 5).

Experiments conducted in [75] demonstrated that the group size (number of consumer-agents) and task number play a role in the level of specialization within the environment. We, therefore, tested our simulations with task number 2, 5 10, 20, 100 and group size of 5, 10, 50, 100, 500, 1000. That is, we tested our model with 2 tasks and 5 agents, 2 tasks and 10 agents, 2 tasks and 50 agents and so on. As we are considering the human social system, we felt that the increased number of tasks and agents in the system would provide a clear impression of agent specialization under varying parameters (threshold, stimulus, demand, etc.).

The proposed model aids agent specialization by modeling the selection of tasks using positive social influence exerted by the neighboring experts. Therefore, the number of experts for each combination of the above-mentioned variation of the number of tasks and agents is to be selected. For a given number of Consumer-Agents, the number of experts in the network is selected as minimum, average, half, equal and maximum value of number of Consumer-Agents. That is, if there are 50 agents (Consumer-Agents) in the system, then we test our agent specialization under the values of varying experts ranging from minimum to maximum. Therefore, the number of experts considered are minimum (~10%), average (~35%), half (~50%), equal (~100%), maximum (~200%). That means the number of experts in the network tested is [5, 17, 25, 50, 100] for 50 agents (Consumer-Agents). Meaning, the model is tested for 50 agents and 5 producer-agents, 50 agents and 17 producer agents and so on.

The combination of task numbers, group size and the varying number of experts in the network is tested. The frequency of the tasks selected by Consumer-Agents for the mentioned setup is recorded. We ran 100 simulations for every combination of task number, group size, and expert number. The frequency of the tasks recorded for each simulation by all the agents is stored in an nxm matrix, where n indicates each agent and m each task. This matrix is subjected to a series of calculations and the DOL value of the matrix is obtained. The quantitative description of the division of labor reflects the extent to which each agent is a specialist. Division of Labor is quantified and the details can be found in [74]. The DOL value results between 0 and 1. A score of 1 indicates that all the agents are fully specialized, while 0 indicates, no specialization. The obtained results are compared with the standard genetic threshold model.

The calculation of DOL value is presented in detail here. Once the frequency of selected tasks is recorded and presented in a matrix, the matrix is normalized. The matrix is normalized by dividing each entry of the matrix by the total frequencies of all the selected tasks, so the sum of all the entries in the matrix equals 1.

The DOL value is calculated by the formula,

1. Division of consumer-agents(X) into tasks(Y) $D_{Y|X}$: $\frac{Mutual\ entropy\ over\ joint\ distribution\ of\ X\ and\ Y}{Shanon's\ index\ (entropy)\ over\ X}$

2. $Mutual\ entropy\ over\ joint\ distribution\ of\ X\ and\ Y$ (I(X,Y)):

$$I(X,Y) = \sum p(x,y) \log\left[\frac{p(x,y)}{p(x)p(y)}\right]$$

$$\text{where, } (p(X,Y), x \in \mathrm{X}, \mathrm{y} \in \mathrm{Y}))$$

p(x,y) is the bivariate probability over a combination of set of Consumer-Agents (labeled X) and set of tasks (labeled Y).

3. $Shanon's\ index$ [81] $(entropy)\ over\ X$, H(X):

$$H(X) = -\mathrm{p(x)log}\sum[\mathrm{p(x)}]$$

4. Therefore, DOL value is calculated using,

$$D_{Y|X} : \frac{I(X,Y)}{H(X)} = \frac{\sum \boldsymbol{p(x,y)} \mathbf{log}\left[\frac{\boldsymbol{p(x,y)}}{\boldsymbol{p(x)p(y)}}\right]}{-\sum \boldsymbol{p(x)} \mathbf{log}[\boldsymbol{p(x)}]}$$

Consumer-Agents and their selected tasks are captured. Now, the Consumer-Agent should acquire the required capabilities to achieve the selected task. The missing required capabilities are listed and are acquired one by one from suitable Producer-Agents using CA. The operational cost, distance cost, and time-taken for execution of the algorithms are captured.

A total of ten capabilities is considered and every Consumer-Agent has a range of three to four missing capabilities. For the missing capability set of each Consumer-Agent, the algorithm is executed for 100 generations/iterations. The missing capability is represented using list in python. The candidate solutions are a vector of the Producer-Agent team represented using a list data structure as well. The length of the list of the Producer-Agent combination team is as same as the length of the missing capability list. The fitness score of the summation of operational cost and distance cost (precomputed distance cost from Consumer-Agent to each Producer-Agent) is assigned to every candidate solution. The top teams of specified number(tp) are updated to belief space. Belief Space is comprised of a list of lists, that is, the main list data structure has sublists of all the selected top teams. A combination of the items in the sublists generates new candidate solutions and the main list is sorted with respect to the fitness score of the sublist candidate solutions. The elite team from the generated candidate solutions from the belief space is added to the new population and the same procedure is repeated for a given number of generations.

We consider two setups to implement and test our proposed CA. In the first setup, the network graph with 50 Consumer-Agents and 100 Producer-Agents is given as input. A population size of 100 is set. The near-optimal solution of suitable Producer-Agent team is identified and their respective operational cost, distance cost, fitness score (sum of operational and distance cost) is captured. The average operational cost, distance cost and fitness score for the entire Consumer-Agent capability requirement is reported. The execution time taken by the algorithm are reported.

Similarly, in the second setup, the network graph with 100 Consumer-Agents and 200 Producer-Agents is given as input. A total of ten capabilities is considered and every Consumer-Agent has a range of three to four missing capabilities. For the missing capability set of each Consumer-Agent, the algorithm is executed for 100 generations/iterations. A population size of 150 is set. The near-optimal solution of suitable Producer-Agent team is identified and their respective operational cost, distance cost, fitness score (sum of operational and distance cost) is captured. The average operational cost, distance cost and fitness score for the entire Consumer-Agent capability requirement are reported. The execution time taken by the algorithm is reported.

The results obtained (average operational cost, average distance cost, and average fitness score) by executing CA for the two setups are compared with the exhaustive search technique, random search technique and GA (Genetic Algorithm). The execution time taken by all the algorithms is reported and compared. The system specifications on which the experiments are performed is as described below,

Processor	Intel(R) Core (TM) i5
Installed RAM	8 GB
System Type	64-bit OS, x64-based processor
OS	Windows 10 V 1903
Version used	Python 3 (Anaconda)

Table 1 System (Device) Specification

Chapter 5: Evaluation and Results

The performance of the proposed model is evaluated, and the results are reported and analyzed in this section. We used MABS to simulate the emergence of agent specialization under the influence of positive social influence exerted by the experts in the system/network. The evolution of Consumer-Agent capabilities is addressed by identifying the suitable Producer-Agents in the network from whom the Consumer-Agents can acquire the required capabilities from, by using an evolutionary algorithm (CA). In our social system, there are Consumer-Agents and Producer-Agents mapped to a network. Each individual in a network, that is, each person in society knows some people, for example, teacher, coach, family members, friends, etc. All the individuals in a network communicate or interact with others who are connected directly or indirectly. Each agent should select a task, which is influenced by the known neighboring experts in the network. On the other hand, an agent has to evolve over time in a system/environment but has some missing capabilities, which needs suitable Producer-Agents to collaborate with in order to acquire the missing capabilities, in turn, achieving the selected tasks.

Each Producer-Agent in the network provide services like teaching, coaching, mentoring, etc., within related operational cost. This operational cost can be financial cost or time. As mentioned, the aim of this dissertation is to find suitable Producer-Agents with minimum operational and distance cost to acquire from and evolve over time and to attain specialization by selecting appropriate tasks. Several scenarios to deal with the problem are simulated and the obtained results are reported. The result of the standard genetic threshold model is reported for comparison. In addition, our Cultural Algorithm (CA) is tested for varying parameters and the results are reported in comparison with the exhaustive search, random search, and genetic algorithm.

5.1 Quantifying Emergence of Specialization

Consumer-Agent selects a task using two methods namely, Standard Genetic Threshold Model (random selection of tasks under no expert influence) and Positive Social Influence Model (selection of tasks under expert influence). In Random Selection Model, an agent selects a task randomly. In Positive Social Influence Selection model, an agent selects a task that is influenced by the agent's network expert neighbors.

To measure and quantify the degree to which the agents are specialized, the chosen specialization of all agents needs to be recorded. This is achieved by having each active agent record its specialization at the end of each iteration. The mutual information and Shannon entropy index for the distribution of individual agents across tasks is calculated. Dividing mutual information score by Shannon entropy score results in values ranging from 0 to 1, indicating how specialized agents are. Score 1 indicates that all the agents (Consumer-Agents) are fully specialized, while 0 indicates no specialization.

For each combination of group size (5, 10, 50, 100, 500, 1000) and task numbers (2, 5 10, 20, 100), division of labor (DOL) is quantified.

The DOL values for the genetic threshold model for the above-mentioned combinations are as shown, and the rows indicate the number of tasks and columns indicate the group size (Consumer-Agents). The cell values show the DOL value with the standard deviation.

GT	5	10	50	100	500	1000
2	0.580 ± 0.120	0.555 ± 0.155	0.617 ± 0.023	0.702 ± 0.099	0.712 ± 0.036	0.719 ± 0.056
5	0.528 ± 0.098	0.559 ± 0.075	0.663 ± 0.026	0.733 ± 0.054	0.740 ± 0.033	0.765 ± 0.050
10	0.469 ± 0.054	0.536 ± 0.094	0.665 ± 0.038	0.778 ± 0.018	0.778 ± 0.042	0.782 ± 0.048
20	0.417 ± 0.063	0.510 ± 0.111	0.702 ± 0.048	0.800 ± 0.015	0.800 ± 0.031	0.800 ± 0.035
100	0.378 ± 0.008	0.428 ± 0.029	0.560 ± 0.029	0.663 ± 0.026	0.823 ± 0.038	0.844 ± 0.050

Table 2 DOL values of Genetic Threshold Model

The demand level considered in our experiment is $\partial = 0.7$, and at the demand level less than one, both group size and task number positively affects division of labor. The detailed analysis of the results of the genetic threshold model can be found in [75].

In our PSI (Positive Social Influence) model, for each combination of group size (5, 10, 50, 100, 500, 1000), task numbers (2, 5 10, 20, 100) and varying expert numbers (minimum, average, half, equal, maximum), division of labor (DOL) is quantified for our Positive Social Influence Model. The model is implemented using “Mesa Agent Based Modeling” and the figure below shows how the parameters can be adjusted using the sliders shown on the left side of the figure.

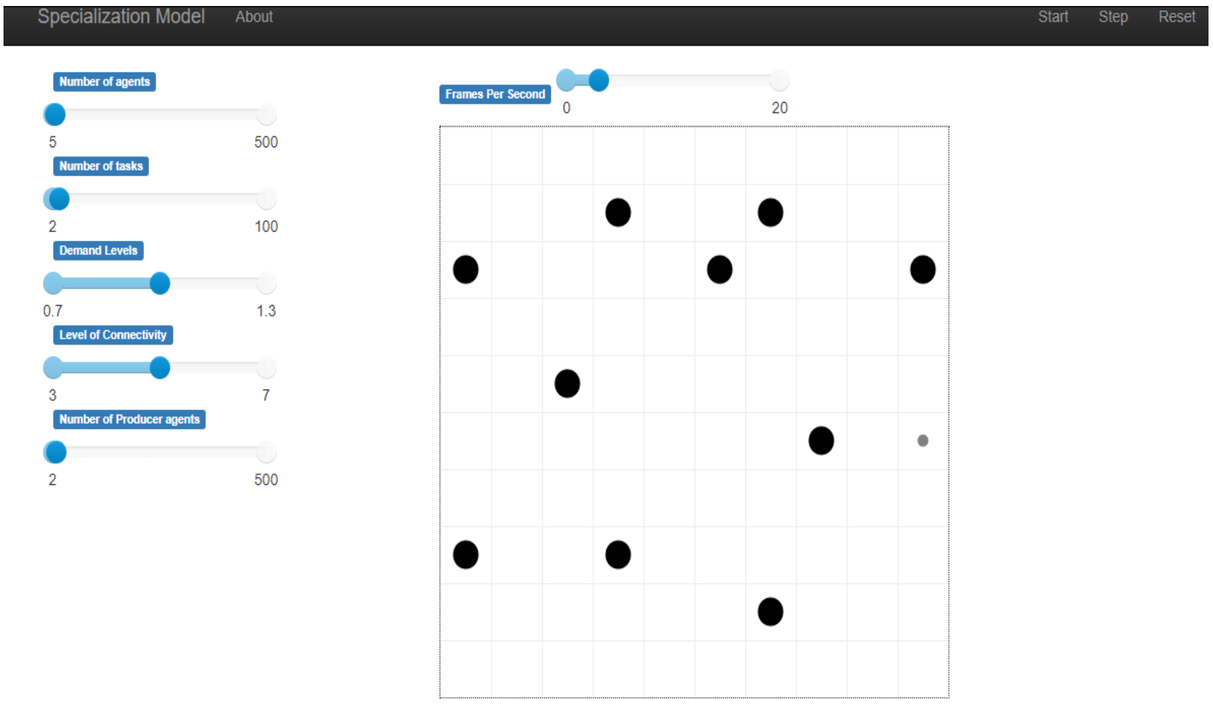

Figure 7 The screenshot of our implemented Specialization Model (PSI)

The grid shows the number of agents in the system. Number of Consumer-Agents, the number of tasks, demand levels, level of connectivity (of the social network implemented using the Barabasi-Albert model) and number of Producer-Agents is adjustable. After providing inputs, the model is started. At the end of the simulation, we obtain the recorded information of tasks/ specialization selected by the agents. Then the frequency of the selected task specializations is presented in a matrix and the dol calculations are performed.

The DOL values for the positive social influence model for the varying parameter combinations are as shown, the rows indicate the number of tasks and columns indicate the group size (Consumer-Agents). The cell values show the DOL value with the standard deviation. %E represents the value of percentage of agents considered and E represents the number of experts.

PSI	**%E**	**E**	**5**	**E**	**10**	**E**	**50**	**E**	**100**	**E**	**500**	**E**	**1000**
	~10%	1	0.793 ± 0.132	1	0.746 ± 0.107	5	0.628 ± 0.022	10	0.727 ± 0.089	50	0.766 ± 0.081	100	0.756 ± 0.047
	~35%	2	0.763 ± 0.143	3	0.760 ± 0.106	17	0.644 ± 0.025	35	0.735 ± 0.084	175	0.754 ± 0.084	350	0.758 ± 0.029
2	~50%	3	0.783 ± 0.139	5	0.765 ± 0.109	25	0.653 ± 0.022	50	0.727 ± 0.087	250	0.751 ± 0.084	500	0.762 ± 0.030
	Equal	5	0.821 ± 0.132	10	0.763 ± 0.094	50	0.665 ± 0.029	100	0.704 ± 0.094	500	0.742 ± 0.082	1000	0.765 ± 0.026
	Max	10	0.782 ± 0.128	20	0.771 ± 0.102	100	0.679 ± 0.023	200	0.706 ± 0.091	1000	0.744 ± 0.081	2000	0.747 ± 0.027
	~10%	1	0.648 ± 0.127	1	0.667 ± 0.071	5	0.640 ± 0.020	10	0.759 ± 0.063	50	0.768 ± 0.048	100	0.773 ± 0.035
	~35%	2	0.674 ± 0.122	3	0.697 ± 0.075	17	0.702 ± 0.053	35	0.750 ± 0.065	175	0.763 ± 0.058	350	0.805 ± 0.027
5	~50%	3	0.674 ± 0.140	5	0.709 ± 0.067	25	0.655 ± 0.022	50	0.748 ± 0.062	250	0.753 ± 0.055	500	0.816 ± 0.023
	Equal	5	0.700 ± 0.122	10	0.702 ± 0.072	50	0.666 ± 0.020	100	0.742 ± 0.074	500	0.745 ± 0.054	1000	0.822 ± 0.026
	Max	10	0.688 ± 0.133	20	0.696 ± 0.065	100	0.678 ± 0.024	200	0.749 ± 0.063	1000	0.731 ± 0.064	2000	0.806 ± 0.025
	~10%	1	0.543 ± 0.051	1	0.634 ± 0.049	5	0.693 ± 0.062	10	0.779 ± 0.065	50	0.765 ± 0.063	100	0.826 ± 0.076
	~35%	2	0.548 ± 0.050	3	0.641 ± 0.064	17	0.744 ± 0.039	35	0.777 ± 0.053	175	0.782 ± 0.065	350	0.869 ± 0.066
10	~50%	3	0.555 ± 0.046	5	0.642 ± 0.049	25	0.726 ± 0.049	50	0773 ± 0.069	250	0.790 ± 0.040	500	0.867 ± 0.061
	Equal	5	0.553 ± 0.049	10	0.638 ± 0.050	50	0.725 ± 0.045	100	0.776 ± 0.067	500	0.786 ± 0.037	1000	0.862 ± 0.060

	Max	10	0.560 ± 0.051	20	0.657 ± 0.058	100	0.733 ± 0.048	200	0.775 ± 0.069	1000	0.782 ± 0.042	2000	0.857 ± 0.067
	~10%	1	0.477 ± 0.025	1	0.575 ± 0.025	5	0.768 ± 0.048	10	0.804 ± 0.054	50	0.845 ± 0.015	100	0.879 ± 0.051
	~35%	2	0.478 ± 0.027	3	0.573 ± 0.024	17	0.763 ± 0.058	35	0.817 ± 0.043	175	0.851 ± 0.045	350	0.905 ± 0.044
20	~50%	3	0.477 ± 0.026	5	0.580 ± 0.028	25	0.753 ± 0.055	50	0.814 ± 0.042	250	0.813 ± 0.042	500	0.911 ± 0.043
	Equal	5	0.479 ± 0.025	10	0.582 ± 0.032	50	0.745 ± 0.054	100	0.820 ± 0.036	500	0.815 ± 0.039	1000	0.910 ± 0.039
	Max	10	0.479 ± 0.025	20	0.580 ± 0.028	100	0.731 ± 0.064	200	0.828 ± 0.041	1000	0.822 ± 0.035	2000	0.894 ± 0.041
	~10%	1	0.378 ± 0.008	1	0.458 ± 0.007	5	0.614 ± 0.034	10	0.722 ± 0.083	50	0.853 ± 0.040	100	0.911 ± 0.042
	~35%	2	0.377 ± 0.008	3	0.459 ± 0.007	17	0.632 ± 0.034	35	0.694 ± 0.080	175	0.865 ± 0.043	350	0.922 ± 0.036
100	~50%	3	0.376 ± 0.009	5	0.460 ± 0.006	25	0.639 ± 0.031	50	0.718 ± 0.080	250	0.852 ± 0.042	500	0.932 ± 0.033
	Equal	5	0.379 ± 0.008	10	0.460 ± 0.008	50	0.649 ± 0.037	100	0.708 ± 0.091	500	0.860 ± 0.042	1000	0.931 ± 0.032
	Max	10	0.379 ± 0.008	20	0.464 ± 0.008	100	0.652 ± 0.035	200	0.733 ± 0.051	1000	0.860 ± 0.038	2000	0.921 ± 0.035

Table 3 DOL values of PSI Model

The reason for varying dol values for varying numbers of experts is discussed here. Though the experts are introduced in the network for a varying percentage of agents ranging from minimum to maximum, the resulting dol values, that is agent specialization selection is influenced by the neighboring experts alone. That is even if there are 20 experts in the network, if the agent has no direct expert neighbors then the agent has no influence at all. Then the agent ends up selecting tasks randomly without any influence exerted by the experts in the environment. Such agents turn up to grow as generalists without a particular specialization.

If the experts are at the neighboring nodes, then there exist four different possibilities in resulting dol values. If there is a single expert in the neighboring nodes which influences one specialization selection, increasing the agent specialization. Multiple experts with similar specialization (same specialization) influence one specialization selection which furthermore increases the degree of specialization of an agent (Consumer-Agent). Complete specialization is a possibility here with resulting dol values of 1. Multiple experts with varying (different) specialization influence various specialization (tasks) selection, decreasing the agent specialization. The agent is confused with multiple specialization options and ends up selecting multiple tasks and grow to be a generalist. Also, at times the filtered task set may not have the specialization influenced by the neighboring experts, then the agent ends up selecting a random task, which reduces the degree of specialization.

In a given system, at a given point in time, there are active and inactive agents. For demand levels less than one, the work available in the environment is less than the total efforts needed from the agents. Therefore, with low demand levels, the inactivity of the agents has no negative impact on the division of labor, because the active agents in the system efficiently perform the tasks. That means, at lower demand levels, only fewer agents are needed to perform the available tasks as the work required (to perform all tasks) is less than the efforts available (from all the agents in the system). Although few agents remain inactive, their presence is not detrimental to task specialization because all work is completed by the active agents at each time step and so the stimulus levels do not accumulate. The variations/fluctuations in the stimulus levels decreases with increased group size and decreased demand levels and hence partially offsets the negative effect of group size on division of labor. It is evident from the dol values in the table, that the ratio of group size and task number significantly influences the dol values. Changes in the task number is likely to be strongest at the stage where the group size is lower. The demand level becomes a stronger influence on dol values as the group size grows. For small group size, the dol values

decreases with the increase in the task number. For increased group size, the lesser demand aids in lower fluctuations of stimulus level, in turn, decreasing the randomness in task selection and increasing the dol values, which is increasing the level of specialization.
Analyzing from both task number and group size variation, for a given task number the dol values increases as the group size increases. The rows from left to right shows that the dol values increases with group size for any given task number. Whereas, the values of the column represent that for the smaller group size the dol values decreases as the task number increases. The first two columns represent this pattern. But as the group size increases, the dol values increases with increase in task number as a result of demand the stimulus levels impact. Group size and task number positively affect division of labor.

By comparing the results of the 'Genetic Threshold Model' and our 'Positive Social Influence Model', that is by comparing the dol values from table 2 and table 3, it is evident that our model results in a significant increase in the dol values, in turn, improving the degree of agent specialization.

5.2 Capability Evolution using CA

Consumer-Agent selects a task using PSI, where an agent selects a task that is influenced by the agent's expert network neighbors. The agent must acquire the missing required capabilities for the task selected.

Each Consumer-Agent has predefined capabilities out of which we formulate few capabilities as available capabilities, and few capabilities as not available capabilities to show the missing capabilities. In order to complete each task, several capabilities are needed. Required capabilities are captured and the Consumer-Agents acquire the capabilities from suitable Producer-Agents. Each Producer-Agent provides some capabilities/services and each capability is associated with an operational cost. The operational cost is assigned by a random number between 0 and 10 (0 <= *operational cost* <= 10).

As discussed, knowledge belief space is used to find suitable Producer-Agents offering the services with minimum operational and distance costs.

The operational cost, distance cost, and execution time taken by our algorithm (CA) is captured and are compared with the exhaustive search technique, random search technique, and GA. The suitable Producer-Agents team for every capability set requirement is identified using all the techniques (all the algorithms) mentioned above. The fitness score gives the summation of operational cost and distance costs. The average of all fitness score, operational cost and distance cost is calculated for the setup implemented and the average values are reported.

We formulate the setup details for two network combinations. The first setup includes a network with 50 consumer-agents and 100 producer-agents. Each consumer-agent has a set of capability requirements for which the suitable Producer-Agent team is to be identified. The number of iteration/generations considered for GA and CA is 100. The frequency of capabilities possessed by Producer-Agents are not taken into consideration. The frequency is random and is not defined. The size of the population is set to 100. The total number of capabilities is set to ten and the missing capabilities range from 3 to 4. That is, the capability requirement set has three or four missing capabilities. In our CA approach, the search space is directed to identify the suitable Producer-Agent team using the knowledge from the belief space. The average fitness score, operational cost, distance cost and time taken by our model for setup 1 is as shown,

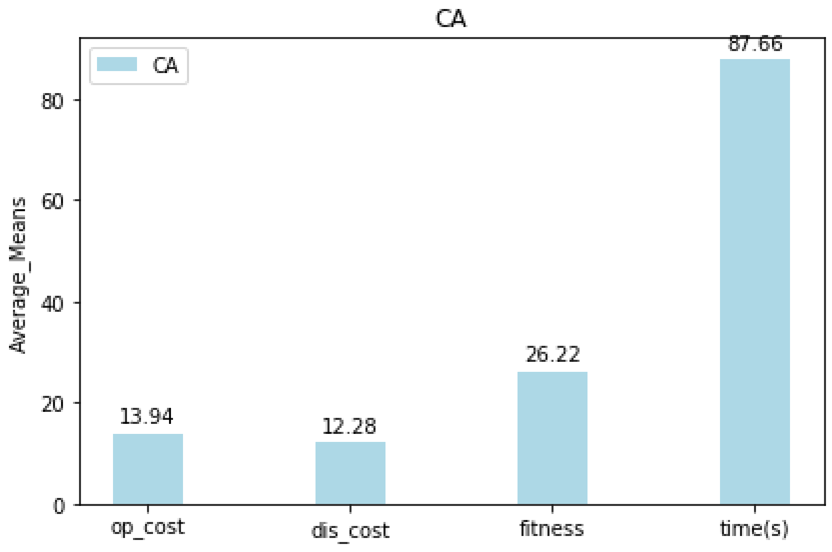

Figure 8 CA Result for 50 x 100 Network

Similarly, for setup 2, 100 Consumer-Agents and 200 Producer-Agents graph is given as input. The setup is similar to the first setup. Whereas, the population size is set to 150. In our CA approach, the search space is directed to identify the suitable Producer-Agent team using the knowledge from the belief space. The average fitness score, operational cost, distance cost and time taken by our model for setup 2 is as shown,

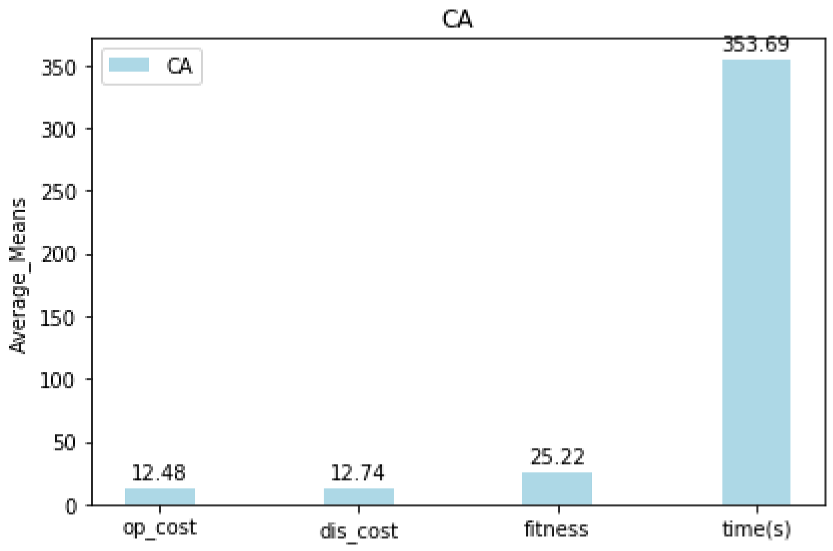

Figure 9 CA Result for 100 x 200 Network

In the Exhaustive search technique, the whole population is analyzed every time for suitable Producer-Agents who offer required capabilities/services with minimum operational and distance costs. Though the suitable Producer-Agent with minimum operational cost and distance cost is found, the time taken by this technique is too high when compared to the proposed model. The results for exhaustive search technique for network combination of 50 X 100 (50 Consumer-Agents and 100 Producer-Agents) is reported below,

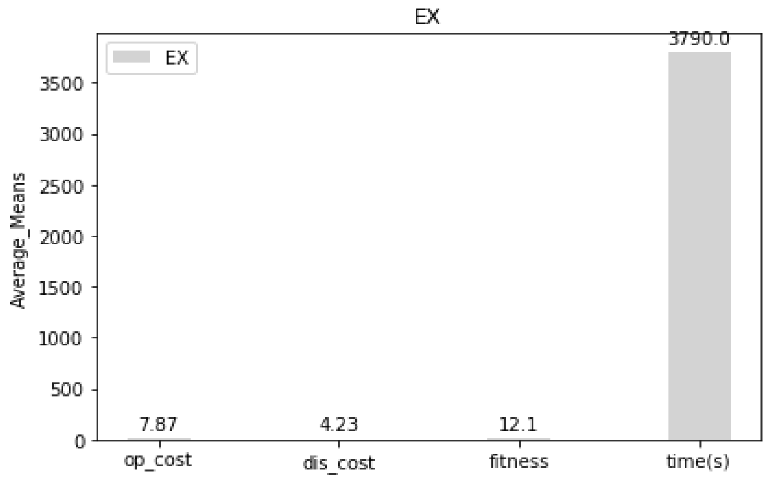

Figure 10 EX Result for 50 x 100 Network

The results for the exhaustive search technique for network combination of 100 X 200 (100 Consumer-Agents and 200 Producer-Agents) is reported below,

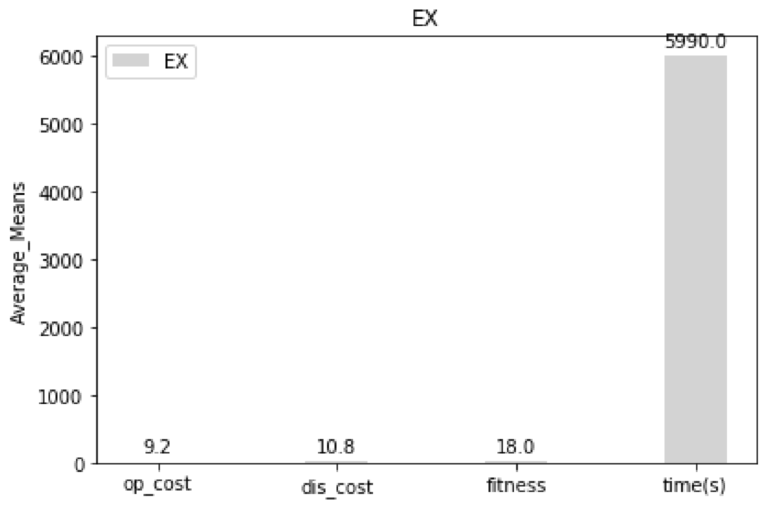

Figure 11 EX Result for 100 x 200 Network

In the Random technique, the Consumer-Agent acquires the capabilities from available Producer-Agents which provides the required capabilities. There is no decision-making process involved in the random approach that is being compared here and hence time taken is relatively lower in this technique. As shown in the figure below, the obtained operational cost, distance cost by random search method is reported. The results for the random approach for network combination of 50 X 100 (50 Consumer-Agents and 100 Producer-Agents) are shown below,

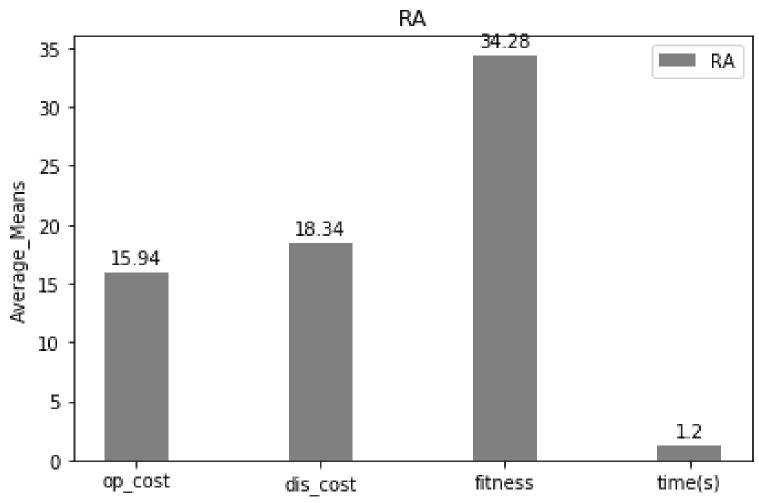

Figure 12 RA Result for 50 x 100 Network

The results for random approach for network combination of 100 X 200 (100 Consumer-Agents and 200 Producer-Agents) are shown below and It is evident that the proposed method outsmarts the random approach

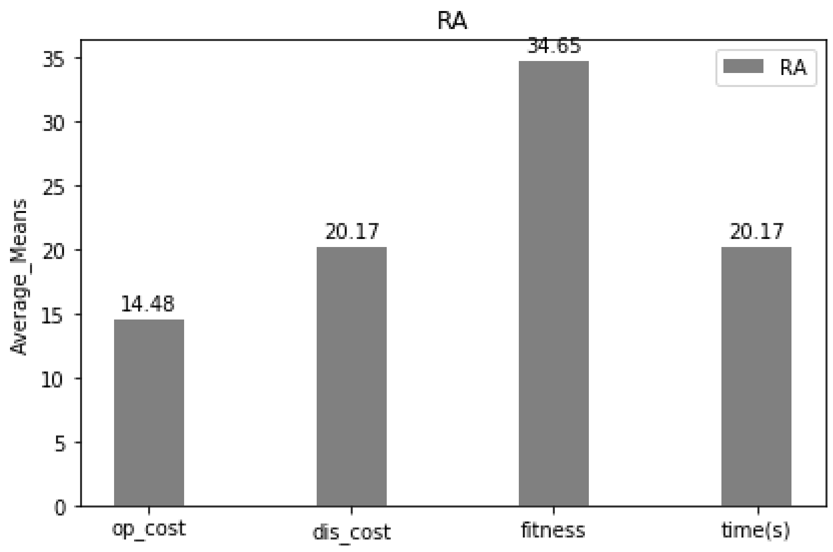

Figure 13 RA Result for 100 x 200 Network

In GA, the same setup as proposed model is used, but a GA component lacks belief space. The results for GA for network combination of 50 X 100 (50 Consumer-Agents and 100 Producer-Agents) are shown below,

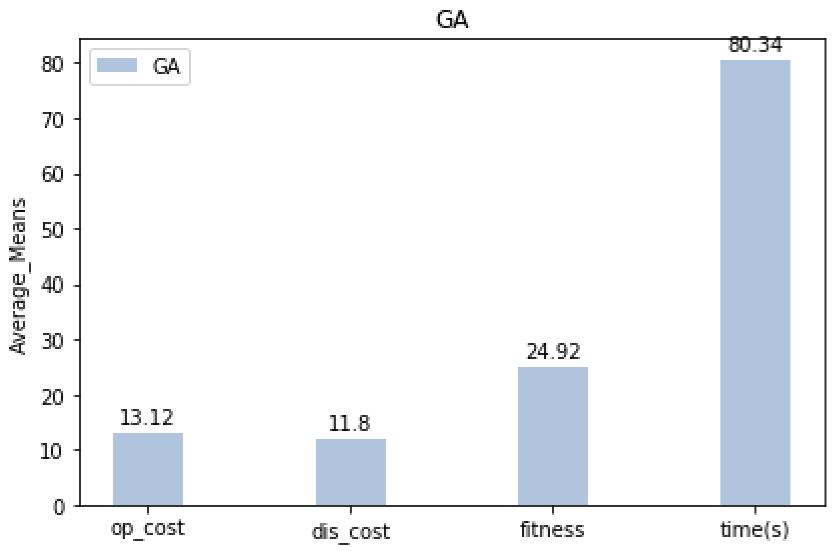

Figure 14 GA Result for 50 x 100 Network

The results for GA for network combination of 100 X 200 (100 Consumer-Agents and 200 Producer-Agents) are shown below,

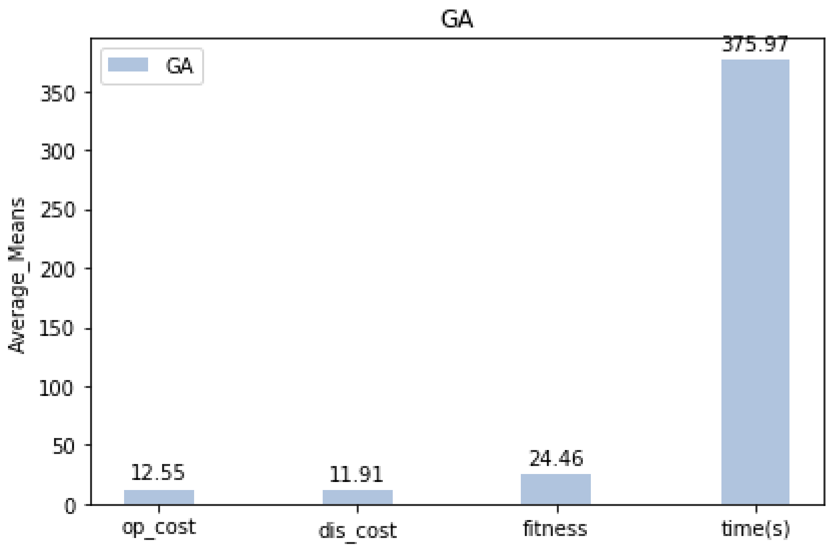

Figure 15 GA Result for 100 x 200 Network

The results (average operational cost, average distance cost, and time taken) of GA are approximately similar to the proposed model.

According to the results, it is evident that the proposed model finds a near-optimal solution at reduced execution time. In comparison with the exhaustive search algorithm, our algorithm finds the near-optimal solutions at significantly lesser execution time. In comparison with GA, no significant changes for the considered setup are observed when the results of CA and GA approaches are compared. In comparison with the random search algorithm, our algorithm reports the near-optimal solution for operational and distance costs, but the time taken by random search algorithm is relatively low as there is no computational process involved in selecting a solution randomly. The accuracy of the results of our approach is significantly feasible when compared with the overall results of all the approaches.

Chapter 6: Discussion

In this chapter, we will be discussing a working example, challenges, and limitations that need to be addressed while implementing our proposed system in the real world.

Let the system under study be a working educational model where the ‘students’ are Consumer-Agents and the Producer-Agents can be ‘teachers’, ‘parents’, ‘guides’, ’university’, ‘coaching centers’ etc. The tasks in the Task-set include, ‘computer engineering’(T_1), ‘sports professional’(T_2), ‘physician’(T_3), etc. Now, the student has to select a task to perform, that is, the student has to select the tasks and should allocate the resources (time) in the respective tasks to perform the tasks.

Consider three students who are directly connected to experts and also note that the task selection is influenced by the direct neighboring experts alone.

Case 1: ‘student 1’ has ‘parent’ as a single neighboring expert. Let the expertise of the parent (expert) be ‘Computer Engineer’. The task ‘computer engineer’ (T_1) will be selected by the student and the student spends most of his/her resource (time) in performing the selected task. And task ‘sports professional’(T_2) might be selected by the student, but a minimum of his/her resource will be spent on task 2 as this task is not influenced by the expert. This scenario contributes to the increase in dol values. Meaning, the student spends more time perfecting a task rather than generalizing in many tasks, therefore, the improved dol values suggest that the degree of specialization of the student is improved respectively.

Case 2: ‘student 2’ has ‘parent’ whose expertise is ‘Computer Engineer’ and ‘teacher’ whose expertise is ‘computer science subject’. The task ‘computer engineer’ (T_1) will be selected by the student and the student spends a maximum of his/her resource (time) in performing the selected task. This scenario contributes to the increase in dol values on a higher degree.

Case 3: ‘student 3’ has ‘parent’ whose expertise is ‘computer engineer’, ‘teacher’ whose expertise is ‘biology subject’ and ‘coach’ whose expertise is ‘basketball’. Now the student has many experts with varying specialization which results in confusion. The student tends to allocate his/her resources among all the three tasks and this scenario decreases the degree of specialization of the student.

Case 4: ‘student’ has no direct neighboring expert. Therefore, a random task is selected the dol value decreases respectively.

Case 5: The filtered task set might not have the specialization influenced by the expert. Therefore, the student selects the tasks from the filtered task set randomly. This scenario results in the decrement of dol values.

These are the five scenarios contributing to the varying dol values in Table 3.

Coming to the ‘capability evolution’ section, case 1 shows that the ‘student 1’ selects $T_{1.}$ The capabilities of acquiring this selected specialization include reqc1: Reading, reqc2: Writing, reqc3: Coding, reqc4: Problem-solving ability, reqc5: Technical skills, reqc6: Research skills. The student has minimal capabilities and the capabilities like coding, technical skills, and research skills are missing from the student’s capability set. Now, with multiple Producer-Agents in the system offering the required missing capabilities, the suitable team of Producer-Agents offering the missing capabilities is identified using cultural algorithms.

Cultural algorithms aids in identifying the best suitable Producer-Agent team offering the services/capabilities at near-optimal operational and distance costs at reduced execution time.

Conclusively, our model guides the student’s decision-making ability in task selection by considering positive social influence exerted by neighboring experts. Additionally, our model aids the students to acquire the required capabilities by identifying the best suitable Producer-Agents in the network (system).

Our model can be adapted to mimic the real-world social systems like society, educational institutions, corporations, etc to improve the productivity of the individuals and in turn improving the productivity of the overall system.

However, there are a few challenges and limitations that need to be addressed in adapting the proposed model to the real-world. The challenges and limitations of implementing the model to a real system are, as listed,

a. The interrelationships between the agents (individuals) of the system are hard to realize, capture, and replicate.
b. The demand factor will not always be a static number. That is, the demand factor can vary with respect to the dynamics of the environment of the considered system.
c. Our model considers the positive social influence exerted by experts alone, but in reality, the agents (individuals) might be subjected to negative social influence as well.
d. Precomputing the distance costs between the Consumer-Agents and Producer-Agents might take up a lot of storage space as the network grows.
e. Currently, our model considers one single network setup for one simulation run, but the real network connecting the individuals mimicking the interrelationships between the agents is a changing network (dynamic network).

Chapter 7: Conclusion and Future work

7.1 Conclusion

The proposed MABS model simulates the social system under study by considering the demand factor, variations in stimulus and varying threshold values to attain specialization by using the benefits of social networks. In our specialization model, we consider the positive social influence exerted by the expert neighbors directly connected with the Consumer-Agents in selecting the specialization (selecting the tasks).

Once the task is selected, the required capabilities to achieve the selected task is to be acquired by the Consumer-Agents. The evolution of agent capabilities is addressed with the identification of best suitable Producer-Agents using an evolutionary component (CA).

The Consumer-Agent aims to acquire the capabilities by searching the social network to find suitable Producer-Agents who provides the services with minimum operational cost and distance costs.

Results (DOL values) show that our proposed model of specialization improves the degree of specialization of agents successfully by considering expert's positive social influence in the network when compared with the results of the genetic threshold model.

According to the results, the proposed model performs better than the exhaustive search model, random selection model, and GA by identifying the near-optimal solution of a suitable Producer-Agent team with minimum operational cost, minimum distance cost at reduced execution time.

The proposed model is an augmented model that supports the decision-making process and aids in improving the degree of specialization by considering positive social influence exerted by agent’s neighboring experts and in identifying the suitable Producer-Agents team to acquire the required capabilities, in turn, achieving the selected tasks.

Limitations of the proposed model include the assumption that all the capabilities are available in the network. That is, all the required capabilities are serviced by the Producer-Agents in the network which is not the case in a real system.

7.2 Future work

In the future, we are going to test the performance of our algorithm on the real-world data and compare them with the actual decisions that a human expert can take manually. Meanwhile, exploring the performance of our algorithm on larger and dynamic networks is another goal. The distance cost precomputed in our model can be efficiently calculated using ‘on the go’ techniques like 2 hop cover [82][83]. The frequency of capabilities serviced by the Producer-Agents can be considered for more efficient modeling.

References

[1] M. Nima, and Z. Kobti, “Extending External Agent Capabilities in Healthcare Social Networks,” Electronic Theses and Dissertations, Windsor, 2017.

[2] M. Felicitas, and Z. Kobti “Modeling the Evolution of Artifact Capabilities in Multi Agent Based Simulations,” Electronic Theses and Dissertations, Windsor, 2015.

[3] D. Cockburn and Z. Kobti, “The Effect of Social Influence on Agent Specialization in Small-World Social Networks,” Electronic Theses and Dissertations, Windsor, 2009.

[4] I. David, S. David, and M. Antonio, “Agents Applied in Healthcare: A review,” International Journal of Medical Informatics, 73(9): 145-166, 2010.

[5] F. Darren, M. Carolyn and E. Samir, “A Survey of Agent Based Intelligent Decision Support Systems to Support Clinical Management and Research,” In Proceedings of the 2nd International Workshop on Multi-Agent Systems for Medicine, Computational Biology, and Bioinformatics, pages 16–34, 2005.

[6] G. Nigel, “Agent-Based Social Simulation: Dealing with Complexity,” The Complex Systems Network of Excellence, 9(25):1-14, 2004.

[7] J. Hong, K.Waldemar, Z. A. Tareq, “Applications of Agent Based Simulation for Human Socio-Cultural Behavior Modeling,” Work: A Journal of Prevention, Assessment and Rehabilitation, 41:2274–2278, 2012.

[8] M. M. Charles, and J. N. Micheal,“ Tutorial on agent-based modeling and simulation,” In Proceedings of the 37th conference on Winter simulation, pages 2–15. Winter Simulation Conference, 2005.

[9] Z. Q. Shen, “Goal-oriented Modeling for Intelligent Agents and their Applications ,” Ph.D. Thesis, Nanyang Technological University, Singapore, 2003.

[10] J. Kim, K. Park, and S.Park, “Agent Identification Method using Goal Modeling,” 2000.

[11] M. Kolp, P. Giorgini, and J. Mylopoulos, “A Goal-Based Organizational Perspective on Multi-Agent Architectures,” In Proceedings of the Eighth International Workshop on Agent Theories. Architectures, and Languages (ATAL-2001), pp. 146-158, 2001.

[12] K. Park, J. Kim, and S. Park, “Goal-Based Agent-Oriented Software Modeling,” In Proceedings of the 7th Asia-Pacific Software Engineering Conference (APSEC'00), Singapore, December 05-08, 2000.

[13] J.B. Larsen, “Specialization and Division of Labour in Distributed Autonomous Agents,” University of Aarhus. 2001.

[14] S.O'Donnell, “Rapd markers suggest genotypic effects on forager specializatoin in a eusocial wasp,” Behav. Ecol. Sociobiology, 38:83-88, 1996.

[15] R.E.J Page, J. Erber, and M.K Fondrk, “The Effect of Genotype on Response Thresholds to Sucrose and Foraging Behavior of Honey Bees,” (apis mellifera I.). J. Comp. Physiol.A, 182:489-500, 1998.

[16] S. J. Russell and P. Norvig, “Artificial Intelligence: A Modern Approach,” New Jersey: Prentice Hall: (2nd ed.), Upper Saddle River, Chapt. 2, 2003.

[17] D. Alexis, F. Jacques, “Multi-Agent Simulation as a Tool for Modeling Societies: Application to Social Differentiation in Ant Colonies,” In Artificial Social Systems, pages 2–23. Springer, 1994.

[18] D. Cockburn, and Z.Kobti, “Wasps: A Weight-Allocated Social Pressure System for the Emergence of Agent Specialization,” European Conference on Artificial Life, pages 161-167, 2011.

[19] D. Cockburn, and Z.Kobti, “The emergence of specialization in heterogeneous artificial agent populations,” ProQuest Dissertations & Theses Global, 2012.

[20] J. R. Zamora, J. Muriciano, and A. Millan, “Specialization in Multi-Agent Systems Through Learning,” Biological Cybernetics, 76(5):375-82, 1997.

[21] A. K. Vera, S. W. Annie, “Specialization versus Re-Specialization: Effects of Hebbian Learning in a Dynamic Environment,” Artificial Intelligence Research Society Conference (FLAIRS-31), 2018.

[22] W. Lee, and D. Kim, “History-Based Response Threshold Model for Division of Labor in Multi-Agent Systems,” Sensors, Basel, Switzerland, 2017.

[23] Z. Kobti, and R.G. Reynolds, “Modeling Protein Exchange Across the Social Network in the Village Multi-Agent Simulation,” In Systems, Man and Cybernetics, 2005 IEEE International Conference on, volume 4, pages 3197–3203, 2005.

[24] R. Nicole, D. Virginia, J Catholijn, A. Theo, and T. Harry, “On the Engineering of Agent-Based Simulations of Social Activities with Social Networks,” Information and Software Technology, 54(6):625–638, 2012.

[25] Z. Kobti, R. G. Reynolds, T. Kohler, "A Multi-Agent Simulation using Cultural Algorithms: The Effect of Culture on the Resilience of Social Systems," In Proceedings of the IEEE Congress on Evolutionary Computation, volume 3, pages 1988–1995, 2003.

[26] Z. Kobti, R. G. Reynolds, T. Kohler, "The Effect of Kinship Co-Operation Learning Strategy and Culture on the Resilience of Social Systems in the Village Multi-Agent Simulation," In Evolutionary Computation, 2004. CEC2004. Congress on, volume 2, pages 1743–1750. IEEE, 2004.

[27] Z. Kobti, A.W. Snowdon, S. Rahaman, T. Dunlop, and R.D. Kent, "A Cultural Algorithm to Guide Driver Learning in Applying Child Vehicle Safety Restraint," In Proceedings of the IEEE Congress on Evolutionary Computation, pages 1111–1118, 2006.

[28] O. Alberto, H. Arturo, M. Fernando, P. Julio, C. Laura, J. Lenka, and L. Liang, "Artificial Societies and Social Simulation Using Ant Colony, Particle Swarm Optimization and Cultural Algorithms," INTECH Open Access Publisher, 2010.

[29] F. Andresen, B. Schulte, and H. Kohler, "Capability Evolution in the Armed Forces: Agents, Practices, and Processes of Dynamic Capabilities," Academy of Management Proceedings, 2019.

[30] T. Parsons, "Social System," Routledge, 2013.

[31] E. Bonabeau, "Agent-Based Modeling: Methods and Techniques for Simulating Human Systems," Proceedings of the National Academy of Sciences, 99(suppl 3):7280–7287, 2002.

[32] V. Julian, and V. Botti, "Multi Agent Systems," 2019.

[33] N. Muaz, and H. Amir, "Agent-Based Computing for Multi-Agent Systems to Agent Based Models: A Visual Survey," Scientometrics, vol. 89(2), pp. 479-499, 2013.

[34] W. Contributors, "Agent-Based Model". In Wikipedia, The Free Encyclopedia. [Online] Available at https://en.wikipedia.org/w/index.php?title=Agent-based_model&oldid=931360124. [Accessed 26 December 2019].

[35] S. Russell, P. "A modern approach. Artificial Intelligence," Prentice-Hall, Egnlewood Cliffs, 1995.

[36] M. J. Wooldridge, "Reasoning about Rational Agents," Intelligent Robots and Autonomous Agents. The MIT Press, Cambridge, MA, 2000.

[37] N. R. Jennings, "On Agent Based Software Engineering," Artificial intelligence, 117(2):277–296, 2000.

[38] P. Busetta, N. Howden, R. R¨onnquis, and A. Hodgson, "Structuring BDI Agents in Functional Clusters," In International Workshop on Agent Theories, Architectures, and Languages, pages 277–289. Springer, 1999.

[39] L. Padgham, and P. Lambrix, "Formalisations of Capabilities for BDI-Agents," Autonomous Agents and Multi-Agent Systems, 10(3):249–271, 2005.

[40] D. L. Acay, G. Tidhar, and L. Sonenberg, "Extending Agent Capabilities: Tools vs. Agents," In Web Intelligence and Intelligent Agent Technology, 2008. WI- IAT'08. IEEE/WIC/ACM International Conference on, volume 2, pages 259–265. IEEE, 2008.

[41] W. Contributors, “Social Network”. In Wikipedia, The Free Encyclopedia. [Online] Available at https://en.wikipedia.org/w/index.php?title=Social_network&oldid=931175799. [Accessed 27 December 2019].

[42] W. Contributors, “Directed Graph”. In Wikipedia, The Free Encyclopedia. [Online] Available at https://en.wikipedia.org/w/index.php?title=Directed_graph&oldid=931299376. [Accessed 27 December 2019].

[43] W. Contributors, “Graph (discrete mathematics)”. In Wikipedia, The Free Encyclopedia. [Online] Available at https://en.wikipedia.org/w/index.php?title=Graph_(discrete_mathematics)&oldid=921577457. [Accessed 27 December 2019].

[44] L. Hamill, and N. Gilbert, “Social circles: A Simple Structure for Agent-Based Social Network Models,” Journal of Artificial Societies and Social Simulation,12(2), 2009.

[45] L. A. Adamic, R. M. Lukose, A. R. Puniyani, and B. A. Huberman, “Search in Power-law Networks,” Physical review E, 64(4):046135, 2001.

[46] R. G. Reynolds, “An Introduction to Cultural Algorithms,” In Proceedings of the third annual conference on evolutionary programming, pages 131–139, 1994.

[47] R. G. Reynolds, “An Adaptive Computer Model of the Evolution of Agriculture for Hunter-gatherers in the Valley of Oaxaca,” PhD thesis, Dept. of Computer Science, University of Michigan, 1979.

[48] R. G. Reynolds and B. Peng, “Cultural Algorithms: Modeling of How Cultures Learn to Solve Problems,” In Proceedings of the 16th IEEE International Conference on Tools with Artificial Intelligence, pages 166–172. IEEE Computer Society, Washington, DC, 2004.

[49] W. K. Robert, R. E. Frew, and H. H. Mendenhall, “Mobile agent docking arrangement for enhancing agent capabilities,” U.S. Patent 6,148,327, issued November 14, 2000.

[50] J. McNaull, J. C. Augusto, M. Mulvenna, and P. McCullagh, “Multi-agent Interactions for Ambient Assisted Living,” 2011 Seventh International Conference on Intelligent Environments, Nottingham, 2011.

[51] R. Xiao, and T. Yu, “A Multi-Agent Simulation Approach to Rumor Spread in Virtual Community,” Based on Social Network, Intelligent Automation & Soft Computing, 17:7, 859-869, 2011.

[52] M. Z. Pooya, “Social Network Analysis using Cultural Algorithms and its Variants,” Electronic Theses and Dissertations, 2017.

[53] I. Younas, F. Kamrani, C. Schulte, and R. Ayani, “Optimization of Task Assignment to Collaborating Agents,” In Computational Intelligence in Scheduling (SCIS), 2011 IEEE Symposium on, pages 17–24. IEEE, 2011.

[54] A. J. Singh, P. Dalapati, and A. Dutta, “Multi Agent Based Dynamic Task Allocation,” In Agent and Multi-Agent Systems: Technologies and Applications, pages 171–182. Springer, 2014.

[55] T. Patrick, N. Salliou, and R. Thomopoulos, “Coupling Agent-Based Models and Argumentation Framework to Simulate Opinion Dynamics: Application to Vegetarian Diet Diffusion,” Social Simulation Conference (SSC 2019). 2019.

[56] J. Szynal, “Behavioral Clues for Forager Transitions in a Fully Tracked Honeybee Colony,” 2019.

[57] L. Panait, and S. Luke, “Cooperative Multi-Agent Learning: The State of the Art,” Autonomous Agents and Multi-Agent Systems, 11(3):387–434, 2005.

[58] C. John, and C. H. Watkins, “Learning from Delayed Rewards,” PhD thesis, University of Cambridge, England, 1989.

[59] C. John, C. H. Watkins, and P. Dayan, “Q-learning,” Machine learning, 8(3-4):279–292, 1992.

[60] A. Bandura, “Social Foundations of Thought and Action: A Social Cognitive Theory,” Prentice Hall series, 1986.

[61] M. Tomasello, “The Cultural Origins of Human Cognition,” Harvard University Press, 2009.

[62] M. Tomasello, A. C. Kruger, and H. H. Ratner, “Cultural Learning,” Behavioral and Brain Sciences, 16:495–552, 1993.

[63] A. K. Gardiner, D. F. Bjorklund, M. L. Greif, and S. K. Gray, “Choosing and Using Tools: Prior Experience and Task Difficulty Influence Preschoolers’,” Tool-use Strategies. Cognitive Development, 27(3):240–254, 2012.

[64] T. Bossomaier, D. Jarratt, M. M. Anver, J. Thompson and J. Cooper, “Optimisation of Client Trust by Evolutionary Learning of Financial Planning Strategies in an Agent Based Model,” 2005 IEEE Congress on Evolutionary Computation, Edinburgh, Scotland, 2005.

[65] A. Husselmann, “Data-parallel Structural Optimisation in Agent-based Modelling,” ACM SIGEVOlution, 2014.

[66] K. Almejalli, “Intelligent Real-time Decision Support Systems for Road Traffic Management: Multi-Agent Based Fuzzy Neural Networks with a GA Learning Approach in Managing Control Actions of Road Traffic Centres,” 2010.

[67] T. Heikkila, and A. K. Gerlak, “Building a Conceptual Approach to Collective Learning,” Lessons for Public Policy Scholars, 2013.

[68] D. Haan, Robert-Jan, C. Mascha, and V. d. Voort, “On Evaluating Social Learning Outcomes of Serious Games to Collaboratively Address Sustainability Problems: A Literature Review,” Sustainability 10, no. 12, 2018.

[69] J. Baird, R. Plummer, C. Haug, and D. Huitema, “Learning Effects of Interactive Decision-Making Processes for Climate Change Adaptation,” Global Environmental Change, 27, 51-63, 2014.

[70] G. Cundill, and R. Rodela, “A Review of Assertions About the Processes and Outcomes of Social Learning in Natural Resource Management,” Journal of environmental management, 113, 7-14, 2012.

[71] C. Ampatzidou, K. Gugerell, T. Constantinescu, O. Devisch, M. Jauschneg, and M. Berger, “All work and no play? Facilitating Serious Games and Gamified Applications in Participatory Urban Planning and Governance,” 2018.

[72] G. Salvini, A. V. Paassen, A. Ligtenberg, G. C. Carrero, and A. K. Bregt, “A Role-Playing Game as a Tool to Facilitate Social Learning and Collective Action Towards Climate Smart Agriculture: Lessons Learned from Apuí, Brazil,” Environmental science & policy, 63, 113-121, 2016.

[73] N. Becu, A. Marion, A. Brice, B. Elise, B. Xavier, D. Etienne, L. Nathalie, M. Nicolas, P. M. Cécilia, and R. Frederic, “Participatory Simulation to Foster Social Learning on Coastal Flooding Prevention,” Environmental modelling & software 98: 1-11, 2017.

[74] R. Gorelick, S. M. Bertram, P. R. Killeen, and J. H. Fewell, “Normalized Mutual Entropy in Biology: Quantifying Division of Labor,” American Naturalist, 164:678-682, 2004.

[75] R. Jeanson, J. H. Fewell, R. Gorelick, and S. M. Bertram, “Emergence of Increased Division of Labor as a Function of Group Size,” Behavioral Ecology and Sociobiology, 2007.

[76] M. David, and K. Jacqueline, “Mesa: An Agent-Based Modeling Framework. 51-58. 10.25080/Majora-7b98e3ed-009,” 2015.

[77] A. Barabasi, and R. Albert, “Scale-free networks,” Scientific American, 288:60-69, 2003.

[78] S. N. Beshers and J. H. Fewell. “Models of division of labor in social insects,” Annu. Rev. Entomol., 46:413-440,2001.

[79] Bourke AFG. “Colony size, social complexity and reproductive conflict in social insects,” J Evol Biol 12:245-257, 1999.

[80] C. Anderson and D. McShea, “Individual versus social complexity, with particular reference to ant colonies,” Biol Rev 76:211-237, 2001.

[81] Shannon CE, “A mathematical theory of communication,” Bell Syst Tech J 27:379-423, 623-656, 1948.

[82] T. Akiba, Y. Iwata and Y. Yoshida, “Fast Exact Shortest-Path Distance Queries on Large Networks by Pruned Landmark Labelling,” SIGMOD. Pp. 349-360, 2013.

[83] E. Cohen, E. Halperin, H. Kaplan and U. Zwick, “Reachability and Distance Queries via 2-hop Lables,” SODA, pp 937-946, 2002.

Appendix

Appendix A

Code: Specialization Model

Mesa Agent Based Modelling has model and server modules. The model.py describes our model, server.py has the configurations and run.py has the code to launch our model. Command to run the model: python run.py

model.py

```
# libraries used

from mesa import Agent, Model
from mesa.time import RandomActivation
from mesa.space import MultiGrid
from mesa.datacollection import DataCollector
import random
import numpy
from functools import reduce
import time
import pickle
import networkx as nx
import matplotlib.pyplot as plt

# threshold variation implementation
def generate_numbers(wanted_avg, numbers_to_generate):
    initial_selection = [random.choice([45,50,55]) for _ in range(numbers_to_generate)]
    initial_avg = reduce(lambda x, y: x+y, initial_selection) / float(numbers_to_generate)
    if initial_avg == wanted_avg:
        return initial_selection
    off = abs(initial_avg - wanted_avg)
    manipulation = off * numbers_to_generate
    sign = -5 if initial_avg > wanted_avg else 5

    manipulation_action = dict()
    acceptable_indices = range(numbers_to_generate)
    while manipulation > 0:
        random_index = random.choice(acceptable_indices)
        factor = manipulation_action[random_index] if random_index in manipulation_action else 0
        after_manipulation = initial_selection[random_index] + factor + sign * 1
        if 45 <= after_manipulation <= 55:
            if random_index in manipulation_action:
                manipulation_action[random_index] += sign * 1
                manipulation -= 5
            else:
                manipulation_action[random_index] = sign * 1
                manipulation -= 5
        else:
            list(acceptable_indices).remove(random_index)

    for key in manipulation_action:
        initial_selection[key] += manipulation_action[key]
    return list(initial_selection)

def threshold_for_all_tasks(task_num,num_con_agents):
    task_threshold = {}
    k = []
    for j in range(num_con_agents):
        g = generate_numbers(50,task_num)
        task_threshold[j] = g
    return task_threshold

```

```
# barabasi albert model for social network creation

def barabasi_albert_graph(n, m, seed=None):
    if m < 1 or m >=n:
        raise nx.NetworkXError("Barabási–Albert network must have m >= 1 and m < n, m = %d, n = %d" % (m, n))
    if seed is not None:
        random.seed(seed)
    G=empty_graph(m)
    G.name="barabasi_albert_graph(%s,%s)"%(n,m)
    targets=list(range(m))
    repeated_nodes=[]
    source=m
    while source<n:
        G.add_edges_from(zip(*m,targets))
        repeated_nodes.extend(targets)
        repeated_nodes.extend(*m)
        targets = _random_subset(repeated_nodes,m)
        source += 1
    return G

# experts task selected randomly
def prod_tasks(ML,T):
    r = list(range(T))
    pro_t = {}
    for i in range(len(ML)):
        ml = ML[i]
        pro_t[ml] = random.choice(r)
    return pro_t

# neighboring experts
def con_neigh_task(nm,G,N,M,T):
    neigh_list = {}
    for v in G:
        neigh_list[v] = [u for u in G[v]]
    ML = list(range(N,nm))
    MT = prod_tasks(ML,T)
    con_nei = {}
    c = {}
    for i in range(N):
        con_n = (neigh_list[i])
        con_n1 = (list(sorted(set(con_n) & set(ML))))
        con_nei[i] = [[q,MT.get(q)] for q in con_n1 ]
        c[i] = [MT.get(q) for q in con_n1]
    return con_nei,c

# PSI model
class PSI(Model):

    def __init__(self, N, T, D, L, M, width=10, height=10):
        super().__init__()
        self.N = N
        self.T = T
        self.M = M
        self.L = L
        self.grid = MultiGrid(height, width, True)
        self.schedule = RandomActivation(self)
        self.thre = threshold_for_all_tasks(T,N)
        with open("consumer_agents.data","wb") as filehand:
            pickle.dump(self.N-1,filehand)
        filehand.close
        nm = self.N+self.M
        self.G = nx.barabasi_albert_graph(nm,self.L)
        self.datacollector = DataCollector()
        self.c_n_t,self.n_t = con_neigh_task(nm,self.G,self.N,self.M,self.T)
        # Create agents
        for i in range(self.N):
            consumer_agents = ConsumerAgent(i,self.thre,self.n_t,self)
            # Add the agent to a random grid cell
            x = self.random.randrange(self.grid.width)
            y = self.random.randrange(self.grid.height)
            self.grid.place_agent(consumer_agents, (x, y))
            self.schedule.add(consumer_agents)

        for i in range(1):
            tasks = TaskAgent(i,T,N,D,self)
        # Add the agent to a random grid cell
            x = self.random.randrange(self.grid.width)
            y = self.random.randrange(self.grid.height)
            self.grid.place_agent(tasks, (x, y))
            self.schedule.add(tasks)

        self.running = True
        self.datacollector.collect(self)


    global u
    u = []
    def step(self):
        frq=5
        self.schedule.step()
        self.datacollector.collect(self)
        new = self.schedule._agents
        neeww = list(self.schedule._agents.keys())
        neeww.pop(0)
        if len(neeww) == 0:
            self.running = False
        else:
            for q in neeww:
                u_id = new[q].__dict__["unique_id"]
                task_id = new[q].__dict__["task_t"]
                u.append([u_id,task_id])
                if new[q].freq == frq:
                    del new[q]

            with open("consumer_agent_ids.data","wb") as filehandle:
                pickle.dump(u,filehandle)
            filehandle.close()
            with open("task_num.data","wb") as fileha:
                pickle.dump(self.T,fileha)
            fileha.close

    def run_model(self,z=5):
        for i in range(z):
            self.step()

# Task and stimulus

class TaskAgent(Agent):
    def __init__(self,unique_id,T,N,D,model):
        super().__init__(unique_id, model)
        self.task_stimulus = [0]*T
        self.T = T
        self.N = N
        self.D = D
    def step(self):
        self.task_stimulus = [x+(3*(self.N/self.T)*self.D) for x in self.task_stimulus]

# Consumer-Agent steps

class ConsumerAgent(Agent):

    def __init__(self, unique_id, thre, n_t, model):
        super().__init__(unique_id,model)
        self.task_t = None
        self.thre = thre
        self.unique_id = unique_id
        self.freq = 0
        self.n_t = n_t

    def move(self):
        possible_steps = self.model.grid.get_neighborhood(
            self.pos, moore=True, include_center=False
        )
        new_position = self.random.choice(possible_steps)
        self.model.grid.move_agent(self, new_position)

    def select_task(self):
        c = list(self.thre[self.unique_id])
        task_stim = self.model.schedule.agents[0].task_stimulus
        aa = numpy.array(task_stim)
        bb = numpy.array([c])
        f=aa>bb
        F=numpy.where(f)[1]
        if len(F)==0:
            self.task_t = None
        else:
            numF = len(F)
            psi = 0.5
            neighs_tasks = self.n_t.get(self.unique_id)
            if len(neighs_tasks) == 0:
                self.task_t = random.choice(F)
                self.freq+=1
            else:
                nei = [x for x in neighs_tasks if x in F]
                if len(nei) ==0:
                    self.task_t = random.choice(F)
                    self.freq+=1
                else:
                    total_p = 1
                    nei_count = [nei.count(x) for x in F]
                    P=[x/sum(nei_count) for x in nei_count]
                    m = P.index(max(P))
                    self.task_t = F[m]
                    self.freq+=1

        if self.task_t is not None:
            self.move()
            self.model.schedule.agents[0].task_stimulus[self.task_t]-=3

    def step(self):
        self.select_task()
```

server.py

```
from mesa.visualization.ModularVisualization import ModularServer
from .model import PSI
from .model import ConsumerAgent
from mesa.visualization.modules import CanvasGrid
from mesa.visualization.modules import ChartModule
from mesa.visualization.UserParam import UserSettableParameter

def agent_portrayal(agent):
    portrayal = {"Shape": "circle",
                 "Filled": "true",
                 "r": 0.5}
    if type(agent) is ConsumerAgent:
        portrayal["Color"] = "black"
        portrayal["Layer"] = 0
    else:
        portrayal["Color"] = "grey"
        portrayal["Layer"] = 1
        portrayal["r"] = 0.2
    return portrayal
grid = CanvasGrid(agent_portrayal, 10, 10, 500, 500)
chart = ChartModule([
    {"Label": "DOL", "Color": "#0000FF"}],
    data_collector_name='datacollector'
)

model_params = {
    "N": UserSettableParameter('slider', "Number of agents", 10, 5, 500, 50, 100,
                               description="Choose how many agents to include in the model"),
    "T": UserSettableParameter('slider', "Number of tasks", 5, 2, 100, 10, 20,
                                description ="Choose how many tasks to include in the model"),
    "D": UserSettableParameter('slider', "Demand Levels", 1.0, 0.7, 1.3,
                                description="Choose demand level"),
    "L": UserSettableParameter('slider', "Level of Connectivity", 5,3,7,
                                description = "Choose level of connectivity"),
    "M": UserSettableParameter('slider', "Number of Producer agents", 10, 2, 500, 50, 100 ,
                                description = "Choose how many producer agents to include in the
model"),
    "width": 10,
    "height": 10
}

server = ModularServer(PSI, [grid, chart], "Specialization Model", model_params)
server.port = 8521
```

run.py

```
from PSI.server import server
server.launch()
```

Vita Auctoris

Name:	RADHIKA JAYARAMAN
Birth Place:	Bangalore, Karnataka, India
Birth Year:	1991
Education:	Bachelor of Engineering (B.E.), 08/2019-06/2013
	Information Science Engineering (ISE)
	Dr. T. T. I. T. College of Engineering and Technology, KGF, Karnataka, India
	(affiliated under *Visvesvaraya Technological University*, Karnataka, India)
	Master of Science (MSc.), 09/2017-12/2019
	Computer Science (CS)
	School of Computer Science
	University of Windsor, Windsor, Ontario, Canada (UoW)