Towards Institutional versus Interpersonal Influences on Role Adoption

Henry Hexmoor
Computer Science & Computer Engineering Department
Engineering Hall, Room 313
Fayetteville, AR 72701
+01-479-575-6197
hexmoor@uark.edu

Subb Rao Pasupuleti
Computer Science & Computer Engineering Department
Engineering Hall, Room 313
Fayetteville, AR 72701
+01-479-575-2420
spasupu@uark.edu

ABSTRACT

In this paper, we examine role adoption that is both under institutionally prescribed role valuation and interpersonal influences. We contrast a few role adoption norms that progressively show benefits of considering other agents in role adoption. .

Categories and Subject Descriptors

H.1. [MODELS AND PRINCIPLES]: General.

General Terms

Experimentation, Theory

Keywords

Role, Institution, Autonomy

1. Introduction

In multiagent systems, a key research area is optimal instantiation of organizations such that groups of individuals work in a coherent environment and best fill the required roles. With the development of multi-agent systems that simulate human behaviors by artificial agents the concept of *role* has undergone a renaissance [1, 2, 4]. Current research in agent role adoption lacks theories that relate them to the sociological structure or the environment where they evolve.

An institution is a generalized form of a set of organizations that have become commonly understood and perhaps legalized [3, 5, 6]. Institutions prescribe values among roles that can be used to derive utility for the agent's considering role adoption. An institution may also prescribe the number of individuals that can adopt each role. In a given organization, roles are filled with agents who occupy roles according to the institutionally prescribed pattern and norms of promotion and demotion we will explore. Agents in an institution must follow or at least consider the values and norms of their institution. On the other hand, agents influence one another, which may affect their performance and in turn this can have a global effect on the organization's productivity [10]. Fritz Heider's *balance theory* [7] suggests propagation of influences among agents. The basic balance theory hypothesis is that people who find themselves in an unbalanced position should change their relations to generate balance. Following this principle, agents should adopt roles so as to increase their balance.

To motivate ideas in this paper, consider the following outline shown in Figure 1. An institution sets forth norms and conventions. An obvious scenario is retail Industry, which sets norms and conventions. The box labeled "Agents" denotes individual agents that are workers or agent proxies for humans. The organizational units are the places that agents use to perform their function. Agents and organizational units use resources (shown in Figure 1). For example, let's consider human-computer interaction as a scenario. Imagine a human user who needs to use a computation device such as a desktop computer and a number of applications. We will consider the human user as the entity setting values and norms and therefore the institution for this scenario. A third-party software provides the user with a package of agents that will learn the user preferences over time and control the desired applications. There might be an initial mapping of agents to applications. The agents must be assigned to applications in order to best fit the user required pattern. Our algorithms (shown in Figure 1 with heavy border box) are designed for to interface agents with organizational units. Agents adopt roles in ways to address the institutionally imposed role orders as well as considering influences among themselves.

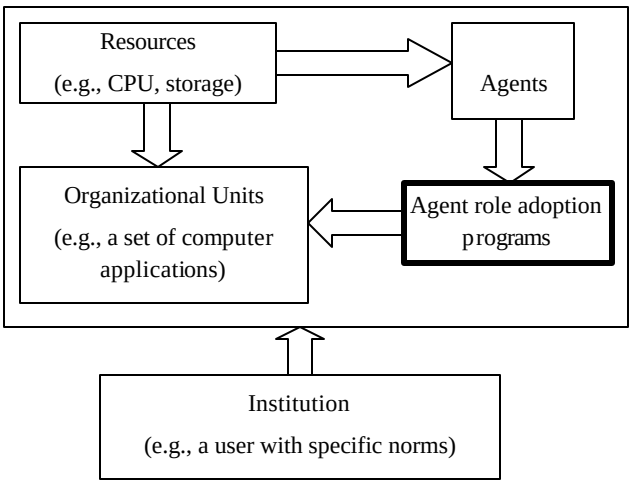

Figure 1. Mapping agents to roles

We introduce a few terms to make our presentation more precise. This precision has been at a cost of simplification of these notions[1].

Definition 1: A *capability* is basic agent ability with a degree in the range from 0.0 to 1.0. We will denote degree of a capability c with D(c).

We assume there is no decay in capability and agents can only increase their capability. Furthermore, we assume capabilities are mutually exclusive. Let C denote a set of capabilities, which are required in the system for performing all tasks. I.e., C = $\{c_1,c_2,\ldots,c_n\}$. C is the set of all capabilities known by all agents. Each agent will possess each capability c_i to a different degree and may improve it by learning. This provides us with an n dimensional space of capabilities. Let's call this a C-space.

Definition 2: A *role*, denoted with r, is a point in C-space that specifies a minimal capability profile to qualify an agent for the role.

For example, with two capabilities c1 and c2, <0.1, 0.5> is a role that an agent to have at least D(c1) > 0.1 and D(c2) > 0.5.

Definition 3: *Rank* of a role r_i assigns a number to the role that reflects its relative importance in an institution. This is denoted by Rank(r_i).

This is a highly simplified model of a role's valuation in an institution. Using this we introduce a notion of role order. The function "Rank" may return any natural number. The smaller the number the more preferred the role. Role r_j is the most preferred rank if Rank(r_j) = 1. Importance of a role is inversely proportional to its rank.

Definition 4: *Role Ordering* (RO) is a total order over roles. Each role is assigned a unique rank. I.e., <Rank (r_1) , Rank (r_2) , ….. , Rank (r_n)> specifies role ordering where Rank(r_i) is the ith position is the rank of ith role.

If Rank(r_i)< Rank(r_j) then role corresponding r_i is preferred over the role r_j that has a smaller rank. RO sets up a trajectory in C-space.

Definition 5: A role pattern, denoted by RP, is a profile of minimum individuals needed to occupy roles in an organization at a given time.

Definition 6: A norm governing role adoption is a set of institutional rules. We denoted the set of norms by RN.

Putting these together, we define an institution.

Definition 7: An Institution is modeled as <C, R, RO, RP, RN>.

In section 2 we describe a few algorithms the implement different role adoption norms. In section 3 we show results of running these algorithms and draw some conclusions in section 4.

2. Algorithms for Role Adoption

When agents are unleashed, each agent adopts a role depending on its capability level. Each agent is generally inclined to move to occupy a role that is higher institutionally ranked than its current role.[2] It will improve its capabilities to qualify for the next higher role. Naturally, at some point in time all agents will qualify for the highest ranked role. Agents will not move up in ranks unless the institutionally specified number of spaces for that role is not yet full. We will call this basic strategy for role adoption **norm 1**. The flow chart in Figure 2 shows our basic algorithm. We will call this Algorithm 1. In section 3 of this paper we will show results of running on this algorithm with percentage of pattern fulfillment over time.

A variation to our algorithm 1 is to include considerations of synergy among agents. Agents may enhance or detract

[1] Since our aim is modeling agents and not human societies we believe this has been an acceptable tradeoff.

[2] We assume each individual belongs to a single institution. In fact, we are considering a single institution for simplification.

from one another's *productivity*. We will define a network among agents with positive and negative influences among pairs of agents.

Definition 8: A *synergy network* among a group of agents is a graph among agents where the arcs represent a real value between -1.0 and 1.0 indicating negative or positive influence between pairs of agents. We will use *s(i,j)* as a symmetric function that returns the synergy value between agents i and j.

As we stated in the start of this paper we want to combine the effects of synergy with capability match to a role. This combination of factors we will call *productivity*.

Definition 9: usefulness(A_i) is a binary function that return 1 when the agent is filling a needed role and 0 if the agent is an extra for that role, i.e., an overflow.

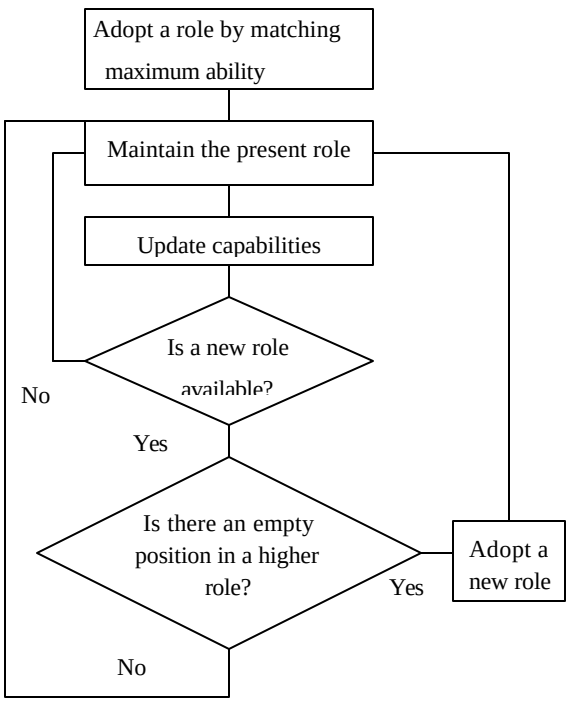

Figure 2. Flow chart for algorithm 1 (norm 1)

An agent is an overflow in a role if there are more than needed agents in that role. This value is 0 if the agent is an overflow or else its value is 1.

Definition 10: *Productivity* of an agent Ai working on role of rank k is a linear combination of (a) average conflict with other agents, (b) role rank desirability, and (C) usefulness.

I.e., Productivity (A_i, R_k)

$$= \frac{1}{\text{sizeof}(r_k)} \sum_{j=1}^{\text{sizeof}(r)} s(i,j) + 1 - \frac{\text{Rank}(r_k)}{\sum_{i=1}^{n} \text{Rank}(r_i)} + \text{usefulness}(A_i)$$

$s(i, j)$ returns synergy value between agents i and j agents. *sizeof(r_k)* returns the number of agents who have adopted role r_k. $\frac{1}{\text{sizeof}(r_k)} \sum_{j=1}^{\text{sizeof}(r)} s(i,j)$ computes the average synergy value of agent i with all other agents who have adopted the same role. $1 - \frac{\text{Rank}(r_k)}{\sum_{i=1}^{n} \text{Rank}(r_i)}$ computes role rank desirability for role k.

Definition 11: *Average Productivity* is the average of productivity (A_i,R_k) over all agents with their adopted role. We will denote this by Average Productivity.

I.e., Average Productivity = $\frac{1}{n} \sum_{i=1}^{n}$ Productivity (A_i, R_k)

Our second algorithm uses the norm of considering individual productivity for deciding to adopt the role. Before adopting a new role, each agent follows a norm of making sure individual productivity is increased. Otherwise, it continues to work on its current role. We will call this **norm 2**, which is a superset of norm 1.

Our third algorithm uses system productivity for deciding to adopt a role. Before adopting a new role each agent makes sure system productivity is increased. Otherwise, it maintains its current role. We will call this **norm 3**. Norm 3 is a superset of norm1 but different than norm 2.

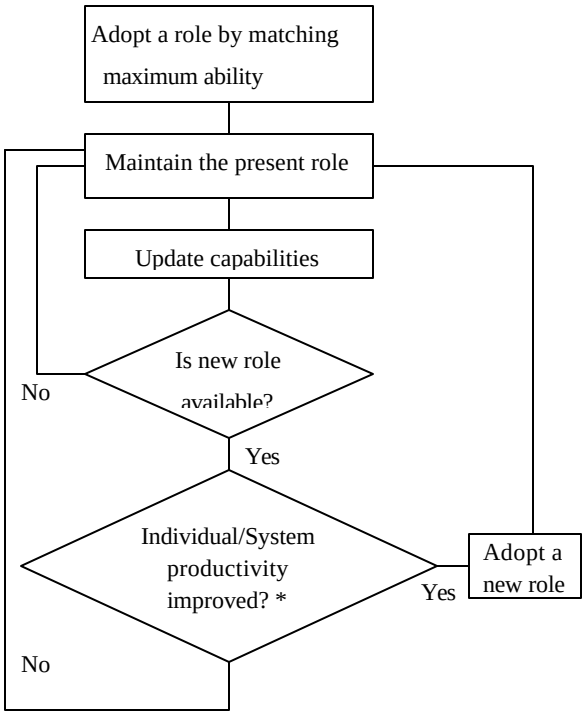

Figure 3. Flow chart for algorithm 2 and 3

*Algorithm 2 considers individual productivity (norm 2), Algorithm 3 considers system productivity (norm 3).

We have used two different approaches in for improving agent capabilities. In the first approach agents learn to improve their capabilities only for roles that are not yet filled in the prescribed role profile. We call agents that use this approach *non-versatile agents*. In second approach agents improve their capabilities irrespective of role profile. Agents which use this approach are called as *versatile agents*. However, agents will not change their roles until it is required by the system (following norm1).

In algorithm 2, each agent adopts a role based on its capabilities. Agents improve their capabilities based on their strategy for capability improvement. Norm 2 specifies that when an agent qualifies for a new role then it will check its productivity by adopting a new role. If it's current productivity can be improved, it will change its role. The difference between the approaches is that an agent only considers its productivity value and it will not consider system productivity.

In algorithm 3, norm 3 specifies that when an agent qualifies for a new role then it will check system productivity by adopting the new role. If that value is more than the present system productivity then it will change its role. The difference between the previous approach and this approach is that an agent considers system productivity.

System productivity can be increased by changes in agent roles to minimize conflict among agents. Algorithms 2 and 3 did not consider this. In algorithm 4, when an agent qualifies for a new role then it will negotiate with all other agents who also qualify for this role and invite them to join into the new role. Each agent who receives this invitation calculates its individual productivity value, if the role productivity value is the highest possible productivity it can produce then it will accept the invitation. This is a variant of contract net negotiation [9]. We used a similar iterative approach for determining agent autonomy [8]. After an agent collects all responses, it calculates the new possible productivity value by considering accepted agents as role candidates for this role. If the calculated value is better than the previous value then it will send invitation to agents who agreed to change their role. We will call this strategy **norm 4**. Norm 4 is different from norms 2 and 3 and a superset of norm 1. This is summarized in the following pseudo-code.

1. Adopt a role by matching maximum ability
2. Improve one of capabilities using learning methods.
3. Check any new role that can be performed.
4. If a new role is available then invite all agents who qualify for this role to join the role.
5. Each agent evaluates this request by calculating its productivity value if it were to adopt a new role. If it improves its productivity it will accept the invitation, else it rejects it.
6. After receiving all agent responses the agent who sent the original request evaluates the new possible productivity value by considering the agents who accepted its request as possible candidates for that role. If newly computed value is better than the current productivity value then the agent adopts the new role and sends that decision to agents to consider changing

Figure 4. Pseudo-code for algorithm 4 (norm 4)

3. Experiments

We conducted a set of experiments with algorithm 1. In an experiment we used 4 roles with ranks <2,4,5,1>. E.g., Role 1's rank is 2, role 2's rank is 4, etc. Role pattern is <7,4,3,4>. E.g., 4 agents are needed for role 2, and 4 agents are needed for role 4. The capability set is $\{c_1, c_2, c_3, c_4\}$. Agents are assigned randomly generated capability levels in each of four capabilities.

Results of this experiment are shown in Figure 5. "Cycles" are simulation cycles. As norm 1 suggests, within each

simulation cycle, each agent improves one of their capability levels and checks whether it attains required capability levels of a new role in the system. We used the same basic setup with the same number of agents, capabilities, and roles in the experiments in remainder of this section.

Randomly generated values between 1.0 and -1.0 are assigned to synergy of agents in a synergy network[3]. In our current simulation, these synergy values are static and do not change. We conducted an experiment on equal number of versatile and non versatile agent groups. Initially, agents in both groups had identical capability levels. Afterwards each group used a different learning strategy.

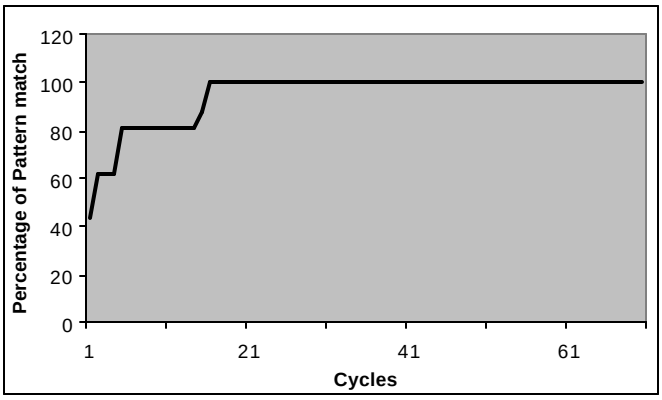

Figure 5. System productivity using algorithm 1 (norm 1)

Figure 6 shows results of an experiment with algorithm 2 using the individual productivity values prescribed by norm 2. Agents only consider their individual productivity in their decision making of adopting a new role. Therefore, as seen in Figure 6, the average system productivity fluctuates[4]. When one agent adopts a new role it may cause other agents' individual productivity to decrease, thereby decreasing the average system productivity. In Figure 6, after 25 cycles the results of learning is evident in rapid frequent average system productivity value changes. This is due to increase number of role changes among agents. Both versatile and non versatile learning strategies gave similar results.

Figure 7 shows the results of an experiment with algorithm 3. Here agents consider norm 3 and pay attention to system productivity values. When we compare this Figure with Figure 6 we observe that the average system productivity value monotonically improves. The average system productivity value becomes constant after reaching the highest possible value. After this value is reached, agents will not change their roles.

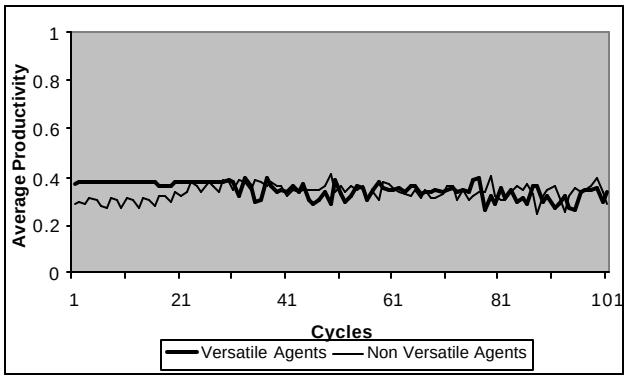

Figure 6. System productivity using algorithm 2 (norm 2)

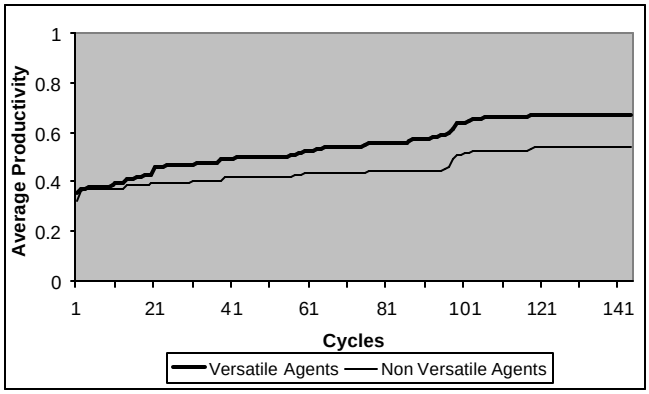

Figure 7. System productivity using algorithm 3 (norm 3)

Figure 7 also shows the difference between the average system productivity values of two sets of agents. In the versatile case, there is always a possibility of learning which improves system value at every cycle. This is not the case for non-versatile agents.

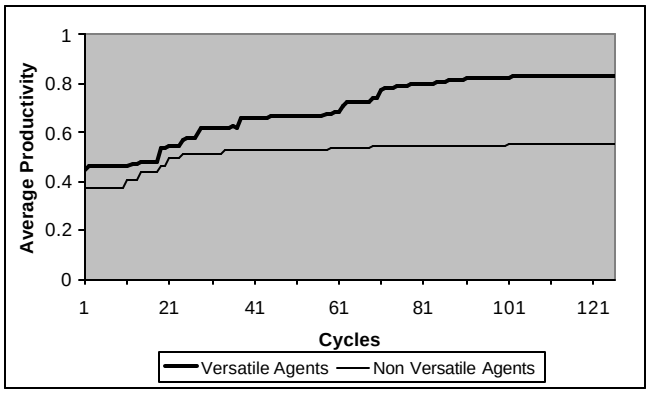

Figure 8. System productivity using algorithm 4 (norm 4)

---

[3] We used the random number generator in Java to generate a number between 0.0 and 1.0. This returns a double value, i.e., 8 point precision. With another call to "random" function in Java we determined a sign for the synergy value.

[4] Since there are three components in the individual agent productivity (see definition 9), we divide that value by 3 in order to normalize it to range of 0.0 to 1.0. This normalization is used for computing the average system productivity as well.

In algorithm 4, agents followed norm 4 and negotiated with other agents when they qualified for a new role. Figure 8 shows results of an experiment with algorithm 4 where agents rapidly improved the average system productivity when compared to algorithm 3. In Figure 7 versatile agents took 50 cycles to achieve average system productivity value of 0.5005, whereas in Figure 8 versatile agents took only 20 cycles to achieve average system productivity of 0.5392. Figure 8 also shows that versatile learning strategy is superior to non versatile learning strategy.

4. Conclusion

We have discussed the contention between institutional ordering of values versus interpersonal influences among agents. We presented a norm that algorithmically combines both effects. We demonstrated the effectiveness of these algorithms with experiments. Versatility in learning capabilities combined with negotiations over potential roles has shown to be the most successful role adoption norm. These techniques are very general and we believe they are applicable to a wide array of problems. For example, a smart home can be treated as an institution where agents can learn preferences of their residents which are role pattern and role preference setters.

We consider three directions for our future work. First, we will explore how agents can quickly adapt to changing preferences in an institution. In the case of a smart home each resident may have a different role order preference requiring agents to adapt quickly. In another extension, we are designing norms that account for resources in role adoption. Lastly, beyond experiments we are working on mathematically contrasting norms.

5. Acknowledgements

This work is supported by AFOSR grant F49620-00-1-0302.

6. References

[1] Cavedon, L., and Sonenberg, L., On Social Commitments, Roles, and Preferred Goals, In *Proceedings of 3rd International Conference on Multiagent Systems*, pp. 80-87, 1998.

[2] Castelfranchi, C. and Falcone, R., From Task Delegation to Role Delegation, In Proceedings of AI-IA 1997: Advances in Artificial Intelligence Congress, LNAI 1321, pp. 278-289, 1997.

[3] Conte, R., Dellarocas, C. (editors), Social Order in Multiagent Systems, Kluwer Academic Publishers, Boston, 2001.

[4] Dastani, M., Dignum, V. Dignum, F. Role Assignment in Open Agent Societies. In: Proceedings of AAMAS'03, Second International Joint Conference on Autonomous Agents and Multi-agent Systems, Melbourne, Australia, July 14 – 18, 2003

[5] Dignum, V. Dignum, F. Modeling agent societies: co-ordination frameworks and institutions, Progress in Artificial Intelligence. P. Brazdil, A. Jorge (eds.), Proceedings of the 10th Portuguese Conference in Artificial Intelligence. Lecture Notes in Artificial Intelligence, Springer-Verlag, Volume 2258, 2001.

[6] Dignum, F. Abstract Norms and Electronic Institutions. In: Proceedings of International Workshop on Regulated Agent-Based Social Systems: Theories and Applications (RASTA'02), at AAMAS, Bologna, Italy, 16 July 2002.

[7] Heider, F. *The Psychology of Interpersonal Relations*. New York: Wiley, 1958.

[8] Hexmoor, H. and Vaughn, J. Computational Adjustable Autonomy for NASA Personal Satellite Assistants, In ACM SAC-02, Madrid, 2002.

[9] Smith, R.G. The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver. IEEE Transactions on Computers 29(12): 1104-1113, 1980.

[10] Zhang, X. and Hexmoor, H. Balancing Individual Capabilities and Social Peer Pressure for Role Adoption, In Proceedings of CEMAS-03, Prague, 2003.