Coalition Formation for Task Allocation via Genetic Algorithms

Mazda Ahmadi  
Computer Engineering  
Department  
Sharif University of Technology  
Tehran, Iran  
m_ahmadi@ce.sharif.edu

Mayssam Sayyadian  
Computer Engineering  
Department  
Sharif University of Technology  
Tehran, Iran  
sayadian@ce.sharif.edu

Hamid R. Rabiee  
Computer Engineering  
Department  
Sharif University of Technology  
Tehran, Iran  
rabiee@sharif.edu

ABSTRACT
In many real world software engineering problems, tasks cannot be accomplished by single agents. In such circumstances, agents have to form a coalition in order to perform the required tasks. In these problems the main concern is to find an optimum coalition structure so that the time for fulfilling the specific number of tasks is minimized. When the number of agents are large, the number of possible coalition structures drastically increases and hence, traditional search algorithms for coalition formation do not work. In this paper we propose to use genetic algorithms to solve this problem. More specifically we introduce our specific genetic based algorithm. To minimize the time for obtaining optimum coalition structure. Finally we will present experimental results to validate our claims.

Keywords: Coalition formation, task allocation, genetic programming

1. INTRODUCTION
Multi agent systems are becoming more important in the emerging software services and applications. One of the main concepts in multi agent systems is cooperation. In many applications even if the agents are selfish, cooperating could be beneficial, because there are many tasks that can be accomplished more optimally by a group of agents [5].

Coalition formation is one of the most widely used methods of cooperation in multi agent systems, and a number of researchers have proposed different algorithms to solve the coalition formation problem [6,7,8,9]. Some of these algorithms guarantee the worst case situation [7] and some do not[6], but all of these algorithms suffer from computational complexity. Specifically, when the number of agents increases the run time of these algorithms increases drastically.

Task allocation is one of the challenging problems in multi agent systems. Different algorithms have been proposed for assigning tasks to agents in order to reach a specific goal [10,11,12].

Genetic based algorithms have been successfully used to search huge spaces[13]. These algorithms never impose a bound on optimum solution but their implementation is easy and they provide satisfactory results in many applications [14].

In this paper, we use a genetic algorithm approach for task allocation and coalition formation. Of course, there is no guarantee for convergence of genetic algorithms but our experimental results have shown that in most cases the result of the algorithm is acceptable. For example, when the number of agents increases so that the traditional algorithms such as those presented in [8] could not solve the problem in a reasonable time, our genetic algorithm is able to solve the problem in an acceptable time.

The paper is organized as follows: In the second section, the environment under consideration is described. In section 3, the problem that we are trying to solve is defined. Section 4 presents the previous works in solving our problem. We introduce our proposed genetic method in section 5. Experimental results are presented in section 6. Finally, the concluding remarks are presented in section 7.

2. PROBLEM SPACE DESCRIPTION
The assumptions and definitions for the specific multi agent system that we are investigating, are described in the following subsections.

2.1 Assumption 1: Rationality
We assume that the agents are group rational. By this we mean that each agent tends to help the whole system achieving more success and obtain more reward. That is when making a decision, an agent may select an action or may participate in a coalition through which he himself receives less reward, in order to increase the reward of the system.
2.2 Assumption 2: Central Decision Making
We assume that a centralized environment is considered, which implies existence of a superior entity in the system. This leads to a central decision making approach. Therefore, there exists an entity in the system to which all agents submit their information. It is through this entity (or the highly authorized agent) that other agents find out what to do or how to cooperate.

2.3 Assumption 3: Conflict
We assume that agents are in mutual conflict with each other. This means using two or more agents together, results in more cost for the whole system. This is independent of the additional value obtained according to the increase in the ability of agents together. This cost could be a result of the need for communication or coordination requested by coalition to survive.

For example in a programming team, if the number of programmers involved increases, the obtained result of the team doesn’t necessarily increase.

2.4 Definition 1: Agent
Agents are defined as entities that have limited resources. An agent owns resources. With using these resources they have to accomplish their assigned tasks. Each agent is characterized by his ability vector, which is a representation of the agent abilities to perform tasks.

2.5 Definition 2: Task
Tasks are defined as pairs of values and resource lists. If all the resources that a task requires are supplied by an agent or a bundle of agents, the agent or the group of agents is capable of performing the task. This way, the resource list of a task characterizes its requisite abilities when supposed to be done according to the decision made by the system.

Each task value is a representation of the reward that is granted to the performer of the task. It means, if all the resources a task needs are supplied by one agent, it is granted the task value as it’s revenue.

If a group of agents supply every needed requirement of a task at least by half, the agents of the group will receive a fraction of the task value proportional to the supplied resources.

2.6 Definition 3: Coalition
Each coalition is defined as a group of agents that are supposed to utilize their abilities jointly to gain more revenue for the whole system. So the coalition can be defined as a pair of objectives and agents. Its constituent parts reflect the characteristics of a coalition. Since each coalition consists of a few agents, so each coalition has an ability vector which is related to the sum of his agents’ ability vectors and a value that represents the whole reward the coalition obtains for the system.

2.7 Definition 4: Rejection
Agents are to some extent at natural conflict with each other. Although it’s beneficial for them to cooperate in order to extend their abilities, but there is a cost for forming each coalition [16,17]. This cost can be seen as a decrease in the ability of a team.

Considering each two agents, they have a rejection value for every resource of their own. Rejection is a tuple consisting of two agents and a list of resource rejections. Taking this into account, the resource list of a coalition is equal to the sum of its engaged agents’ resources minus agents’ pair wise rejections.

2.8 Definition 5: Value of Coalition Formation
The value of a coalition is defined as the sum of its goal values that are supposed to be achieved when forming the coalition. The aim of coalition formation is to find the coalition structure with the maximum value.

3. PROBLEM DEFINITION
The problem is to find the coalition formation with the maximum value, but the search space is huge. If we consider $n$ as the number of agents and $m$ as the number of tasks, the number of possible coalitions becomes $2^m - 1$. But the number of coalition structures is even more. The number of coalition structures is [1]:

$$\Sigma_{i=1}^n Z(a, i)$$

where $Z(a, i)$ is the number of coalition structures with $i$ coalitions. $Z(a, i)$ is the Sterling number of the second kind. It is computed by the equation:

$$Z(a, i) = iZ(a - 1, i) + Z(a - 1, i - 1)$$

When considering that, tasks should be allocated to these coalitions the number of possibilities become $m^a$. Searching in such a huge space is almost impossible in a reasonable time. Although some heuristics have been offered to prune the search space, still the search in these spaces are very time consuming and complex.

4. PREVIOUS WORKS
Different algorithms have been proposed to solve the problem of task allocation with teams of agents and coalition formation. We present previous works in three different groups. The first group has focused on coalition formation methods, the second on task allocation, third on task allocation via coalition formation.

4.1 Coalition Formation
Usually coalition formation is studied in characteristic function games (CFGs) [6,7,8,9]. In such games, value of each coalition is given by a function $v_i$. The $v_i$ shows the reward that this coalition can gain from the system...
while solving the system problem. In many algorithms some minimum value is considered for \(v_e\), so that different type of bounding is possible.

Although coalition formation methods do not consider task allocation, but if the value of coalition structures be somehow related to tasks, these methods can also be used for task allocation (of course such methods are not optimal for task allocation).

\[ \text{Figure 1: Coalition structure graph for a 4-agent system} \]

The coalition structures’ search space is shown by a graph. In this graph, nodes represent coalition structures and arcs represent mergers of two coalitions (see Figure 1). For searching this graph, traditional search algorithms such as bfs or dfs [9] are very time consuming. Different methods have been proposed for optimal search of this graph [9]. Among these we discuss two widely used algorithms.

4.1.1 The Graph Based Method:
This method have been proposed by Sandholm et. al.[7]. Among methods that guarantee a bound, this algorithm is proved to be the best. If the number of agents is equal to \(a\), they have proved, with searching the bottom two levels of this graph \(2^{a-1}\) nodes, that a bound equal to \(a\) could be achieved. That is, if we find a coalition structure with value \(V_m\) the following equation stands for every other coalition with value \(V\):

\[ V_m > a * V \]

They have also proved no other search algorithm (than the one that searches the bottom two levels) can establish any bound \(k\) while searching only \(n = 2^{a-1}\) or less nodes. They lower their bound by searching more levels of the graph. But if we consider the number of agents to be 1000, achieving a bound with \(\frac{1000}{7}\) to optimum needs \(2^{999}\) nodes to be searched. It has been shown that in order to have a low bound on optimum answer, a large number of nodes have to be searched.

4.1.2 The GA Based Algorithm:
Although genetic algorithms haven’t been used much for coalition formation, but Sandip Sen and Sarathi Datta have used it for coalition formation in CFGs [6]. They have set a constraint for coalition values. They assumed that they know the desired size of the coalitions, and relate coalition formation value to this parameter. It seems having such a constraint on coalition formation value is not suitable for some problems. In fact, when agents are heterogeneous and have different abilities, some agents can be worth twice the others. Therefore, in such situations no desired size of coalitions can be considered.

4.2 Task Allocation
Task allocation can be described by traditional problems called set partitioning problems. The formulated form of set partitioning problem is as follows: Given a set \(N = \{ A_1, ..., A_n \} \) and a set of subsets of \(N\), \(S = \{C_1, ..., C_m\}\), such that \(C_j \subseteq N\) and \(\sum S \subseteq S\), such that \(U_{C_j \in S} C_j = N\). The members of \(S\) are the covering sets. If the members of \(S\) are also pair wise disjoint, then \(S\) is a set partitioning of \(N\). We assume that each \(C_j \in S\) has a positive cost \(c_j\). The set-covering problem entails finding the cover with the minimum cost [8].

Some heuristic and greedy algorithms have been proposed to solve the set covering problem by different researchers. For example, Chvatál have succeeded to gain a logarithmic ratio bound [18]. Many researchers have tried to use different genetic algorithms based methods for solving set partitioning problem [19,20].

Although many advanced algorithms have also been presented for set partitioning problem, but none of them consider the cost of forming a team of agents.

Some mathematical algorithms have been proposed specifically for task allocation. For example in [15], Price have used a graph which represents modules and their communication while a multiprocessor array network is represented by another graph. This algorithm tries to match the adjacent matrices to these graphs through sequences of pairwise interchanges of modules. These task allocation algorithms do not also consider the coalition formation costs and teamwork realities between agents.

4.3 Task Allocation via Coalition Formation
Shehory have used Chvatál’s set covering algorithm for a distributed coalition formation method for task allocation [19,20]. Their algorithm consist of two main stages:

1. In the preliminary stage of each algorithm, all possible coalitions are distributively calculated and their initial values are computed.
2. The main stage of the algorithm consists of an iterative distributed greedy procedure in which two sub-stages occur:
   - The coalitional values are re-calculated.
5. PROPOSED METHOD
In this section, we present our approach in using genetic algorithms with real genes to evolve coalition structures. As mentioned above, our search space is the number of possible coalition structures. Furthermore, we assume a centralized approach.

In the following subsection we first describe the semantics of real-valued genes as coalition connections. Coalition connection is a representative of the coalition that contains the agent. Then we present the evaluation function for chromosomes in the GA population.

5.1 Encoding of the Problem
The choice of an efficient representation is one of the most important issues in designing a genetic algorithm. In our GA, each chromosome represents a set of agent to task assignments. That is, each gene represents an agent and the task to which the agent is assigned. This way, for each agent a coalition connection is presented via his gene. The chromosome consists of all agent-task pairs. The first element of each pair - the agent - is fixed. But the characteristic trait of the agent - his assigned task - is possible to evolve. This manner of representing coalition connections and characteristic traits allows us to easily calculate the coalition structures and to simplify the exploration of the search space. It also obviates the need to address coalitions independent of their goal, that is tasks they are assumed to carry out. We use a fixed length chromosome for the genetic algorithm, where each chromosome is made up of a fixed number (n) of coalition connection structures encoded as above.

5.2 Fitness Function Design
Given an appropriate encoding of the problem, the next step in the design of the genetic algorithm is the design of the fitness function. The fitness function evaluates a chromosome in the population. The GA uses this information during reproduction to evolve better chromosomes over generations. Since each chromosome in our representation comprises all agent to task allocations, hence according to the definition of a coalition value, when calculating the fitness function, we first find the coalitions structures that are formed by means of classifying agents according to their allocated tasks, and then the fitness function measures the collective effectiveness of all coalitions according to their coalition value. A coalition is evaluated by how important its designed task is and how much it costs to gather all agents together in a coalition according to their pair wise rejection. The fitness of a particular chromosome is simply the sum of its coalition values.

5.3 Crossover and Mutation
The purpose of recombination operators such as crossover and mutation in a genetic algorithm is to obtain new points in the search space. Since our chromosomes are composed of homogeneous alleles, it suffices to use standard operators. The operators we use are uniform crossover, and swap mutation.

5.3.1 Uniform Crossover:
This operator is one of the standard operators for crossover. The GA uses this operator to generate new generations from previous chromosomes. As stated before, since chromosomes represent agent to task allocations, this operator randomly selects some agents from one of the parent chromosome and replaces his tasks with the other parent's tasks. This way the operator generates a child for the first parent and the same is done to generate the second child but this time the parents are displaced. This operator generates new coalition structures from the already formations. As a matter of fact this operator simply displaces some agents in some randomly chosen coalitions.

5.3.2 Swap Mutation:
This operator is mainly applied to avoid being entrapped in local maxima. It simply selects two agents as parents and displaces their relative tasks.

6. RESULTS
We now report our experimental results using our genetic based method. Our test environment consists of several agents with a random list of abilities. It also consists of some tasks with random values and random requirements. Eleven constants are considered in our test environment which are listed in the following table:

The values between these constant bounds have been generated randomly.

We've tested our method with the given values. Each test has been done 10 times and the average performance is measured. The tests have been done using an Intel 1.8 Giga byte CPU. The results are shown in table 2.

The results show the power of our algorithms in the situations that number of agents grow. In many situations a performance of 56% is achieved.

7. CONCLUSION AND FUTURE WORKS
Coalition formation and task allocation are key points in multi agent systems. In this paper genetic algorithms are proposed for task allocation with help of coalition formation. We investigated other algorithms and showed their weakness when the number of agents grows. Although genetic algorithms does not provide any bound, but it scales up well with the problem size.
We intend to test different heuristic search techniques such as D* and best first search in the proposed coalition structures graph.

Also we are planning to adopt our genetic algorithm method to distributed systems where agents are autonomous and choose their actions on their own, where the rewards should be shared between agents of the coalitions.

8. REFERENCES


conference on Designing the future June 1996 Volume 26 Issue 4


