A Task Oriented Multi-Agent Negotiation Model Based on Contract-Net

Xiang Lin [1],
College of Information Engineering, China Jiliang University, Hangzhou, Zhejiang, China,
Email: xianglin.cjlu@gmail.com

Haijun Tao
College of Information Engineering, China Jiliang University, Hangzhou, Zhejiang, China,
Email:hjtao@cjlu.edu.cn

***Abstract*—Negotiation is one of the key issues in Multi-agent system. In terms of task characteristics, on the foundation of the multi-agent negotiation model based on the acquaintance model, a task oriented extended contract-net negotiation model is proposed after analyzing the advantage and disadvantage of the classical contract-net negotiation model. Based on the model's architecture definition, a formalized coalition definition based on task and acquaintance model and coalition formation algorithm is given, and then an extended contract-net negotiation model based on task coalition is constructed. A combination of GA and simulated annealing algorithm is used to optimize the allocation speed and quality of task coalition in the agent coalition. Experiment shows that the presented model can effectively reduce the negotiation cost and improve the efficiency of task allocation on the basis of ensuring the negotiation quality.**

***Index Terms*—multi-agent negotiation, task model, coalition, genetic algorithm, simulated annealing**

I INTRODUCTION

R. Davis and R.G. Smith proposed a classical multi-agent negotiation model in 1980[1]. It mainly focused on how to allocate the system tasks in order to solve the conflict of resource and knowledge. There are some disadvantages in the classical contract-net negotiation model, mainly exists that system load will rise rapidly with the issue's scale, negotiation quality and change of the task environment can't be guaranteed and adapted, etc. So it has been widely extended in actual applications, such as Sandholm introduced the concepts of hiberarchy and boundary cost computation[2][3]; Fischer introduced the mechanism of temporal trust/refuse and simulation trade in contract-net to optimize task allocation[4]; Collins introduced arbitration mechanism in negotiation process to prevent cheat in bidding[5], etc. In addition, Bouzouia tried to use learning mechanism to optimize task allocation in contract-net[6].

Coalition mechanism is one of the key research issues in MAS negotiation, and it can be used to solve complex problems and reduce negotiation cost. The combination of contract-net and coalition mechanism helps to improve the huge negotiation cost when contract-net processes large scale negotiation tasks. There are many researches about Agent coalition, and researchers analyze the strategies of coalition generation and maintenance in different views. Hu shangli and Rahwan presented the optimized coalition structure generation algorithm[7,8], and Zhang Xinliang and Shi Cunyi presented the coalition structure dynamic generation algorithm[9], which used coalition structure pruning to limit coalition search space efficiently based on the income independence of coalition cooperation. Michalak[10] presented a coalition algorithm based on filter rules to reduces the intractability of the coalition structure generation problem. Aimed at different application areas, there appears some optimized models, such as Xiaowei focused on the utility based optimal task scheduling problem in order to make the task scheduling be more close to actual process[11]; Jiang Jianguo, Zhang Guofu and Xia Na solved optimized generation problem of multi-task coalition from ant colony algorithm and particle swarm algorithm[12,13] and Walaa H proposed a fuzzy-based decision maker negotiation model for coalition formation within a fully decentralized multi-agent system[14]. Mukun presented the decision-making process in agent negotiation, proposed three conception models to support both goal-directed reasoning and reactive response[15]. Bailing proposed a new coalition formation mechanism based on adjust policies based on trust evaluations to enhance the negotiation efficiency[16]. These researches focus on the search strategy of possible coalitions, seek for optimized coalition structure by various approximate algorithms, and consider the generation strategy of coalition little. Combined the coalition negotiation architecture, using the characteristic of coalition architecture to optimize coalition generation is researched little too.

The extended contract-net negotiation model based on acquaintance coalition reduces negotiation cost[17], but when task attributes change greatly, coalition structure hardly keeps stable. Aimed at this disadvantage and task character, this paper formed Agent coalition based on

1 Corrsponeding author

tasks associated with acquaintance model. The formalized coalition definition is given first and coalition generation and maintenance method is designed associated with the acquaintance model. On this foundation, the extended contract-net negotiation model based on task coalition is constructed. Inside the coalition, a combination of GA and simulate annealing algorithm is used to optimize task allocation, which improves the efficiency of task allocation and reduces communication cost[18]. At last, by testing an example of a missile defense system, compared with the traditional contract net algorithm and the contract net model based on acquaintance coalition, it is proved the negotiation model's validation.

II. Architecture of the Negotiation Model

In terms of the distributed application field of the project background and the requirement of the extended contract-net protocol, an architecture of the extended contract-net MAS negotiation model based on task coalition and genetic algorithm is designed as shown in figure 1.

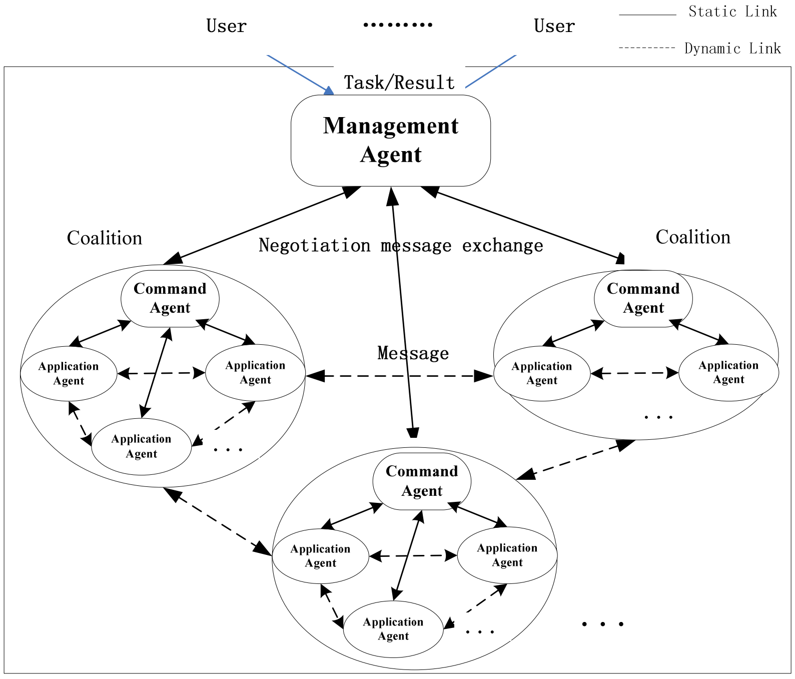


Fig.1 Architecture of negotiation model based on task coalition and GA

According to the requirement of the tasks based on the negotiation of the coalition definition, task coalition becomes the basic task request/assign element in the architecture. In terms of every agent's status in the architecture, agents in the model are categorized as follows. Management Agent (MA) is responsible for interacting with users, receiving tasks from users, decomposing and scheduling them, and returning the results of the tasks to users. Command Agent (CA) is responsible for the organization and management of the members in the agent coalition, the request of the tasks of the coalition and the task allocation in the coalition, i.e. as a leader and a communicator of the coalition. Application Agent (AA) is responsible for detailed tasks, and implements the tasks under the management and cooperation of CA. The structures of TA and CA are specified in details hereinafter (The structure of AA is relatively simple, and has not the tight relation to this negotiation model, so it is not introduced in details later.)

A Management Agent

Management Agent locates in the central position in this negotiation model, and is the logical manager of the model. Figure 2 shows the structure of MA.

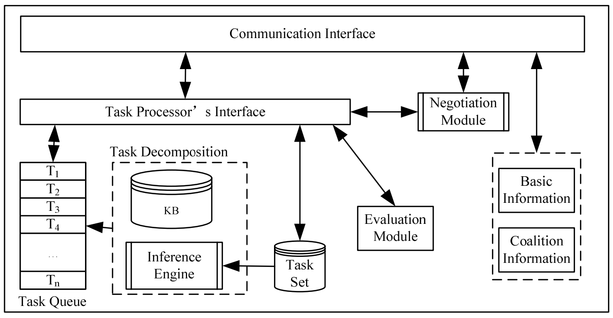


Fig.2 Structure of management agent

The main function of MA is to receive tasks from users and cooperate to deal.

Communication interface is responsible for the message communication between agent and agent coalition. The interface defines FIPA-ACL message syntax, which is used to express negotiation information and task specification, including message storage, decomposition, send and receive, makes agents to communicate seamlessly.

Information module mainly consists of two parts, basic information module and other information module. Basic information module mainly includes various information of MA, and other information module mainly includes the address and status of CA.

Task decomposition module is responsible to decompose tasks input from outside. Its structure includes task decomposition knowledge database, task decomposition inference engine and task set. Knowledge database provides domain knowledge needed by decomposition inference for inference engine. The task decomposition rule table of domain knowledge is the kernel of the knowledge database. Adopting E-WFL inference engine system developed by our group, and changing the rule table and fact table in terms of domain knowledge, the problem of task decomposition has been solved well.

In terms of the capability of every task coalition in MAS, Negotiation module dispatches tasks in the task queue to various task coalitions adopting the task allocation algorithm based on generic algorithm presented hereinafter.

B Command Agent

Command agent is the most important part in the whole model as the coalition commander. The structure of CA is shown in figure 3.

The main functions of Command agent consist of reply task bulletin of MA, send task bulletin to AA according to the tasks received to form coalition, decompose tasks in charge, dispatch tasks charged by the coalition using MAS task allocation algorithm based on generic algorithm, evaluate the results returned from AA and return the results to MA.

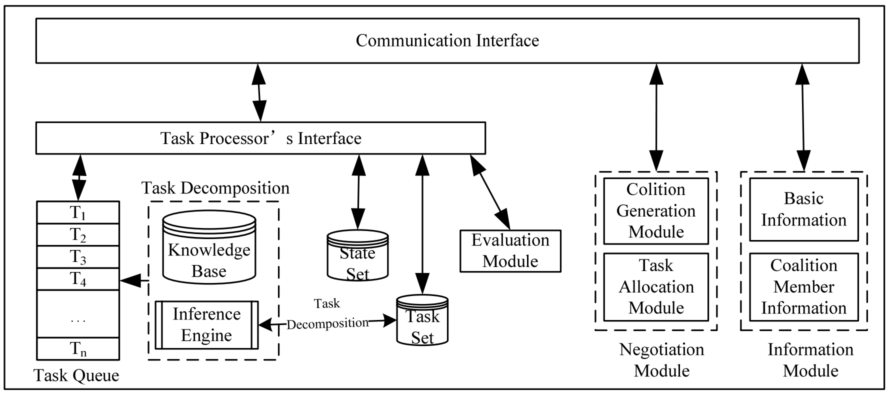


Fig.3 Structure of command agent

Task decomposition module decomposes tasks charged by CA into smaller granularity sub-tasks and stores them in the task queue, and dispatches sub-tasks to AA in the coalition using MAS task allocation algorithm based on generic algorithm. The knowledge database used to decompose task is different to that of MA.

Information module mainly consists of two parts, i.e. basic information module and coalition member information module. Basic information module consists of the basic information, such as CA's address, type, and coalition capability specification, etc. Coalition member information module includes the information of all AA's type, status, and capability in the coalition.

State set stores CA's dynamic information and the intermediate result from sub-task.

Negotiation module is responsible for generating the coalition and dispatching sub-tasks in the coalition. Coalition generation module sends task bulletins to all AAs, receives task request from AA in the limited time, evaluates the request and determines the partner of the coalition, and sends the notice accepted the request to the partners. Task allocation module is responsible for sending task bulletins to AA in the coalition, receiving the replies from AA and the sub-task set completed, coding the AA which requests tasks and the sub-task set the AA can implement, adopts generic algorithm to code in terms of the task requirement, acquires the scheme with the lowest cost to implement the task and sends the task to AA according to the scheme.

III Task Coalition Definition and Generation

On the basis of acquaintance coalition, coalition mechanism based on tasks dynamically adjusts the coalition structure to bid in terms of every task attribute, which makes coalition generation more pertinent to improve problem solving efficiency, and reduces coalition generation cost effectively to improve system reaction speed.

Agents in the system are classified to three types: management agent (MA), command agent (CA) and execute agent (EA).

A. Coalition Definition

1. Capability and Load of Agents

All agents in the system compose a set $A=\{a_1,a_2,\cdots,a_n\}$, every a_i in the set has its capability vector $B_{a_i}=(b_1^{a_i},b_2^{a_i},\cdots,b_r^{a_i})$, and the cost vector $COST_{a_i}=(\cos t_1^{a_i},\cos t_2^{a_i},\cdots,\cos t_r^{a_i})$ agent needs to expend corresponding to capability set. $b_j^{a_i}$ denotes the quantitative capability when a_i can accomplish some specified function, and $\cos t_j^{a_i}$ denotes the cost a_i expend when accomplish $b_j^{a_i}$.

$Load_a \in [0,1]$ denotes Agent a's load, the value is greater, the agent's load is higher, which indicates the agent isn't suitable to accept more tasks.

2. Coalition and its Capability

Coalition is an agent set $C = \{a_1, a_2, \cdots, a_m\}$ which can cooperate to accomplish a task. The commander of the coalition is the manager responsible for the unification action of the whole coalition, and controls the correlative information, such as the member agent's capability in the whole coalition, etc.

The capability of a coalition C is denoted by a vector $B_C = (b_1^c, b_2^c, \cdots, b_r^c)$. The capability of coalition C is the sum of every agent's capability in the coalition, $B_c = \sum_{i=1}^{m} B_{a_i}$.

3. Task and its capability requirement

There are independent tasks $T = \{t_1, t_2, \cdots, t_m\}$ in the system, the priority of task t is denoted by set PRI_t, the priorities of all tasks in the coalition compose set $PRI_T = \{pri_{t_1}, pri_{t_2}, \cdots, pri_{t_m}\}$. The capability needed to solve task t is denoted by $B_t = (b_1^t, b_2^t, \cdots, b_r^t)$, and the essential condition of coalition C can take charge of task t is $B_t[i] \le B_C[i], i \in \{1,2,\cdots,r\}$.

4. the Acquaintance Degree of Coalition C to Agent

The degree of acquaintance denotes the acquaintance degree of the coalition to some agent a_i, and adjusts the members in the coalition by this mechanism. There are two adjustment rules.

①degree of acquaintance increase: if a_i accomplish the task, then $R_{a_i} = \min(1, R_{a_i} + \delta)$, $\delta \in (0,1)$ calls the cooperate factor.

②degree of acquaintance decrease, there are two situations:

- If a_i not accomplish the task, then $R_{a_i} = \max(0, R_{a_i} - \delta)$, $\delta \in (0,1)$ calls the punish factor.
- If a_i not take part in the task, then $R_{a_i} = \max(0, R_{a_i} - \lambda)$, $\lambda \in (0,1)$ calls the forget factor.

The manager of the coalition maintains the current coalition C and the candidate acquaintance set AQ_Set. The candidate acquaintance set stores the agent which attended the coalition, but the current degree of acquaintance is low and hasn't existed in the coalition. The agent in AQ_Set is free and can join in the other coalitions.

According to the degree of acquaintance, coalition adjustment rules are defined as follows:

- rule to remove a member: If $R_{a_i} < R_{Thresh}$, then remove a_i from coalition, R_{Thresh} is the remove threshold. If $R_{a_i} \ge R_{aq}$, then put a_i into the AQ_Set.
- rule to add a member: If a_i matches the requirement, then add a_i to the coalition, $R_{a_i} = R_{init}$, R_{init} is the initial degree of acquaintance.

5. the Degree of Trust

The degree of Agent a_i trusts the other Agent a_j can accomplish the task calls the degree of trust, denoted by $Tr(a)$, $Tr(a) \in [0,1]$. $Tr(a) = 0$ denotes a_i thinks a_j can't accomplish the task.

When bidding object set is large, trust threshold T_l is adopted to control bidding range to improve the efficiency of task consign and to reduce the system cost. The degree of trust is one of the important evaluations when consign tasks. The degree of trust to some Agent/Coalition is higher, the possibility to consign task to it is bigger

When Agent a succeeds to accomplish task t, manager increases $Tr(a)$ by δ_T, $\delta_T \in [0,1]$, denotes:

if $evl(a,t) \ge V_H$, then $Tr(a) = \min[1, Tr(a) + \delta_T]$, V_H is the evaluate threshold of task accomplishment.

When Agent a fails to accomplish task t, manager decreases $Tr(a)$, denotes:

if $evl(a,t) \le V_H$, then $Tr(a) = \max[0, Tr(a) - \delta_T]$.

6. the Computation of Coalition Bidding Value

Coalition needs to compute its bidding value before bidding. The bidding value is computed on the base of analysis to the concrete task, so coalition needs to simulate to decompose and assign to evaluate the bidding value before bidding for some task. The process of the bidding value computation is complex and increases the computation cost when bidding, but the computation process can be stored and used when task in actual execution if hitting.

Assuming the current coalition bidding task is T, coalition decomposes the task into $T = \{t_1, t_2, \cdots, t_n\}$ before it gives the bidding value.

Assuming the sub tasks solved by the coalition itself is $T_{in} = \{t_{i1}, t_{i2}, \cdots, t_{il}\}$, the corresponding cost is $C_{in} = \{c_{i1}, c_{i2}, \cdots, c_{il}\}$, the sum of the cost to solve these sub tasks is $C_n = \sum_{\cos \in Cout} c_{os}$.

Assume the sub tasks solved with the help of other coalitions is $T_{out} = \{t_{o1}, t_{o2}, \cdots, t_{om}\}$, the corresponding cost is $C_{out} = \{c_{o1}, c_{o2}, \cdots, c_{om}\}$, the sum of the cost to solve these sub tasks is $C_n = \sum_{\cos \in Cout} c_{os}$.

Because the coalition bidding task is $T = T_{in} + T_{out}$, and the bidding value of the task T is $V_C(T) = C_{in} + C_{out}$ given by the coalition, it integrates the coalition's capability and the actual situation in the current market, and makes the bidding result accord to the rule of human activities.

B. Generation and Maintenance of Task Coalition

Initial task coalition is generated in terms of the tasks arrived at the first time, during the later procedure, the coalition adjusts continuously according to the degree of acquaintance to agent, and comes to stable gradually. Every time organize coalition in terms of tasks, the coalition manager not only considers candidate acquaintance set, but also considers other agents in the system, and introduces new agent to the coalition to prevent coalition obsolescence.

The method to generate initial task coalition is described as follows:

- Step1: Task manager receives task T, broadcasts the task to all CA in the system, and gives the bidding time limit $Time_T$, waits for CA bidding;
- Step2: CA computes its coalition capability B_c, and compares with B_T:
  - If $B_c \geq B_T$, then computes coalition bidding value $V_C(T)$, and bidding to the manager, goto step3;
  - If $B_c < B_T$, then sends invitation to all Agent $a \in AQ\_Set$, at the same time sends invitation to m Agent $a \notin C \cup AQ\_Set$ randomly, sets reply time $Time_{reply}$;
  - In $Time_{reply}$, CA receives each a_i 's reply, if $a_i \notin AQ\_Set$, set $R_{a_i} = R_{init}$, adjust a_i 's capability value as $B^{'}_{a_i} = R_{a_i} \times B_{a_i}$, record a_i 's load $Load_{a_i}$, form candidate Agent set $Q = \{a_1, a_2, \cdots\}$;
  - CA follows member select strategy, choose appropriate Agent from Q to join the coalition, goto step2;
- Step3: The end.

According to different application areas, this paper defines two kinds of member join strategies:

- the largest capability strategy: if $Count(B_T[i] \leq B_a[i]) \geq T\_Count_{thresh}$, then Agent a decides to join, T_Count_{thresh} is a positive integer;
- average capability strategy: if $\sum_{i=1}^{r} B_a[i] > \eta * \sum_{i=1}^{r} B_T[i]$, then Agent a decides to join.

Coalition member select strategy is similar to coalition join strategy, coalition manager chooses appropriate agent to join the coalition in terms of the capability which the coalition lacked of.

C. Hitting Decision Function

The traditional hitting decision function F adopts different strategies in terms of different application areas, weighted average method, maximum/minimum method and maximum density method (dichotomy method) are some typical methods.

In traditional decision function, the degree of trust between agents is not taken into account. Considered the Agent's degree of trust defined previously, manager modifies the bidding value of Agent a_i to task T_j is:

$$V^{'}(a_i, T_j) = (1-\lambda)V(a_i, T_j) + \lambda * Tr(a_i) * V(a_i, T_j) \quad (1)$$

$\lambda \in [0,1]$ denotes the weight of the bidding agent's degree of trust in the procedure of bidding decision, and hitting agent is filtered according to the modified bidding value. In terms of different application areas, it is commonly taken $\lambda \in [0,0.2]$.

D. Self-Adaptive GA-Based MAS Sub-task Allocation Algorithm

Task allocation in the coalition is an important problem in the negotiation model. It aims to minimizing the cost of allocation. A GA-Based task allocation algorithm is presented to solve the task allocation problem in coalition. This algorithm can solve the task optimized allocation problem better than traditional algorithms.

Task allocation is a typical set coverage problem, and a non-optimized GA not only has a slow convergence speed, but also prones to produce a premature convergence when applied in task allocation. These features have a bad influence on the efficiency and effect of task allocation. Therefore, we use the optimal initial colony selection, optimal parent crossover, Metropolis rule and multi-point crossover/ mutation to optimize the GA according to the application feature.

Set sub-task set is $T = \{t_1, t_2, \cdots, t_n\}$, coalition is $C = \{a_1, a_2, \cdots, a_{l+1}\}$.

（1） Pre-process

Scan all sub-tasks, if Agent a_k is the only agent to accomplish sub-task t_i, then allocate t_i to a_k, and delete t_i from T; if $\exists t_i \in T$, $\neg\exists a \in C, B_a \geq B_{t_i}$, then t_i needs outer cooperation (see negotiation model step10 later in the paper), Do not consider t_i when coding, record the simplified task set is $T^{'}$.

At the same way, scan all Agents, remove the Agent which can't execute any task, i.e. $A^{'} = \{a \mid a \in C, \neg\exists t_i \in T, B_{t_i} \leq B_a\}$, record the simplified Agent set is $C^{'} = C \setminus A$.

（2） Coding Method

CA generates a stochastic binary coded allocation string according to all T_a, called M:

$p_1^1, p_2^1, \cdots, p_n^1, p_1^2, p_2^2, \cdots, p_n^2, \cdots, p_1^l, p_2^l, \cdots, p_n^l$, $l = |C| - 1$ (except CA). M is coded by binary coding, denoted by 0 or 1 (If Agent a_k can't execute t_i, set p_k^i as 0, else set p_k^i as 0 or 1 by stochastic. One task can only be allocated to one agent and every achievable sub-task must be allocated when stochastic allocated.) M is coded to the binary string with the length of $l \times n$.

（3） Crossover Operator

Because each task can only be allocated to one Agent, crossover is bounded by every task allocation string, and isn't allowed crossover arbitrarily, i.e. the string is divided into $A_{t_1} | A_{t_2} | \cdots | A_{t_n}$, and $A_{t_i} = p_i^1 p_i^2 \cdots p_i^l$, crossover can only select the poison of "|".

Choose two pairs of members from the previous generation stochastically, preserve the member with larger fitness in each pair, and generate a stochastic number with uniform distribution between [0, 1]. If S < Crossprobability, then they multi-point crossover uniformly to generate two new members, and use Metropolis rule to the new members and the preserved previous generation member in order to determine whether to put the new member to the next generation. If S > Crossprobability, then the two previous generation members don't crossover, and put them to the next generation directly.

（4） Mutation Operator

Mutation operation adopts multi-point mutation. When p_i^k is mutated, if p_i^k changes $1 \to 0$, it means a_k is deprived the right of execution to t_i, because all tasks should be allocated, it is needed to seek $A = \{a \mid a \in C, a \neq a_k, B_a \geq B_{t_i}\}$, choose a_o from A stochastically, set $p_i^o = 1$.

（5） Fitness Function

$f(a) = \sum Cost_T^m$ ($\sum Cost_T^m$ is the total cost defined by string m at allocation).

（6） The Termination Condition of the Algorithm

The algorithm termination condition is judged by the conjunction of the following two ways:

1) reach specified iterative times K;
2) the degree of fitness matches the following condition:

$$\exists m \in M, f(m) \leq \lambda \times \sum_{i=1}^{n} \min\{Cost_{a_k} \mid t_i \in T, a_k \in \{a \mid B_{t_i} \leq B_a\}\} \quad (2)$$

$\lambda \in (0,1]$ is a constant defined according to the application. The formula means the task allocation solution acceptable as long as the execution cost the solution defined by string m is lower than a proportion of the lowest possible execution cost.

After the algorithm ends, it is needed to evaluate the members with the highest degree of fitness to choose the final allocation solution in terms of load. This paper chooses the member with the largest $\frac{f(m)}{Load_C^m}$ as the final allocation solution, $Load_C^m$ is the sum of the total loads of all Agents in the allocation solution defined by string m.

The optimized GA algorithm is described as below:

```
• The task manager CA in the coalition decomposes the received task $T$ to
  $T = \{t_1, t_2, \cdots, t_n\}$, distributes task invitation to all AA in the coalition,
  and sets the reply limit $Time_{reply}^T$.
• The execution Agent $a$ in the coalition C detects the task invitation, then sends
  task sub-set $T_a = \{t_1^a, t_2^a, \cdots, t_m^a\}$ it can accomplish to CA, i.e.
  $B_{t_i^a} \leq B_L, i \in \{1,2,\cdots, r\}$, and the corresponding cost vector $Cost_a$.
• Pre-process before coding;
• CA codes in terms of the left task sub-set $T'$ and Agent sub-set $C'$, and generates
  $Colony(init)$, $|Colony(init)| = n$.
• Set the selecting threshold $k$, choose $k$ members with the highest degree of fitness
  from $Colony(init)$, choose $l-k$ members from the left elements stochastically,
  compose the initial colony $Colony(0)$.
• Set the iterative number $j = 0$;
• $while$ the appropriate allocation solution not found
    ◆ $j = j+1$, $n = 0$;
    ◆ Put the member with the largest degree of fitness in $Colony(j-1)$ to $Colony(j)$
    ◆ $n = n+1$;
    ◆ $while \ n < l$
        ➢ Choose two pairs of members from $Colony(j)$ stochastically, choose the
          members $a,b$ with the higher degree of fitness in each pair;
        ➢ Generate the uniform distribution stochastic number $S$ between $[0,1]$;
        ➢ $if \ S < Crossprobability$
            ▪ The members $a,b$ multi-point crossover to generate members $c,d$,
              use the following Metropolis rule on $a,c$ and $b,d$ (assume $x$ denotes
              the parent, $y$ denotes the successor) :
              (1) If $f(y) > f(x)$, put $y$ to
```

$Colony(j)$;

(2) If $f(y) \le f(x)$, generate a uniform distribution stochastic number ξ between $[0,1]$, if $\xi < \exp\left[\frac{f(y)-f(x)}{Temperature}\right]$, then take y, else put x to $Colony(j)$;

else put a,b to $Colony(j)$;

➢ $n = n + 2$;

◆ Apply multi-point mutation operator on every member in $Colony(j)$, use the same Metropolis rule on the mutated members;

- Recalculate $Temperature$ according to the temperature decent policy, begin the next iteration;
- Choose the final allocation solution according to the load, return the task allocation solution, the algorithm ends.

Here we adopt a uniform temperature decent policy, i.e. $t_{k+1} = \lambda * t_k$, $\lambda \in (0.5,1)$, select a constant close to 1 in general.

IV Extended Contract-Net Negotiation Model based on Task Coalition

The traditional contract-net model is reconstructed by using the task coalition mechanism and GA-based task allocation algorithm. Coalition is the basic bid unit. The negotiation process is shown as below:

(1) MA receive task TG and stores it in task queue.

(2) When TG is the head of queue, TA check out TG and decompose it into separable sub-tasks: TG= $\{T_1, T_2, \cdots, T_n\}$. Set TA as the system's manager.

(3) If TG is the first task of the system, execute the initial coalition generation according to 3.2. Goto 13;

(4) The manager queries the agent coalition in the blackboard, for each $T_j (j = 1,2\cdots,n)$, send bid invitation message to the active CAs according coalition state and capability information in the blackboard, set the reply time $T_{time-out}$.

(5) According their capability, coalitions that received decides bidding or not. If agent believe that it can execute T_j, then calculate the biding value $V(i,T_j) = b_i$, send the biding value to manager. Otherwise, send invite to the free AAs in the system. If the new coalition can fulfill T_j's require, send bid information to manager, else tell the manager that it can't finish the task.

(6) If the manager receive no tender of T_j in $T_{time-out}$, goto 12, else select a CA as the winner according to award function F, bid value $V'(a_i, T_j)$ and send award to it.

(7) The winner sends confirmation to manager if it plans to execute the task. If the manager doesn't receive the confirmation in $T_{time-out}$, goto 11.

(8) The manager send T_j to the winner, monitor the executing state;

(9) CA invokes the task decomposition module, decomposes T_j to $T_j = \{t_{j1}, t_{j2}, \cdots, t_{jm}\}$, then invokes the task allocation algorithm to allocate the sub-tasks in coalition. When all AAs return result, report the task result to manager. According to the task executing state, regulates the belief value of AA, adjusts the coalition's members;

(10) If the coalition didn't finish all the sub-tasks, the left tasks is called T_j', let $T_j = T_j'$, CA becomes the new manager, goto 11, else goto 13;

(11) The current negotiation round is finished.

(12) The manager decides whether to begin a new bid round. If yes goto 4;

(13) Negotiation process ends.

V Experiments and Analyses

Automatic missile defense system is a typical MAS. It consists of a command center, radar system, several campaign armies. Every campaign army consists of several battle units. Each battle unit consists of several aerial defense missiles. A prototype of the system is designed to test the performance of our negotiation model.

In the system, MA corresponds to the command center. CA corresponds to a campaign army. AA corresponds to a battle unit. The system includes one MA, six CA and thirty AA. The AA's capability consists of attack type, attack range, attack precision, can be described as a five dimension vector *(type,latitude,longitude, range,precision)*, such as $(air,100,85,100,0.8)$, which means attack the target with the center of longitude 100, latitude 85, range 100 kilometers, and the precision is 200.

Three models are involved: traditional contract net model (CNM), acquaintance coalition-based contract net model (ACM) and task coalition-based contract net model (TCM).

Experiment 1: Stable Task Attribute, i.e. invariant invader category

Through the change of task amount, the negotiation average cost and final task solving cost of CNM, ACM and TCM are analyzed. The curves of average optimizing time of the three models are gained by the average value of 10 tests.

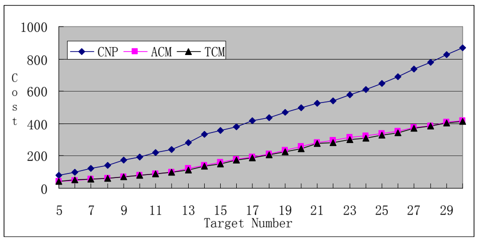


Figure 4. Comparison of negotiation cost

From figure 4, when task attributes change little, task coalition-based and acquaintance coalition-based negotiation costs are close, they both outperform the traditional contract net model, and the negotiation cost of TCM is slightly lower than that of ACM because of the existence of the acquaintance list.

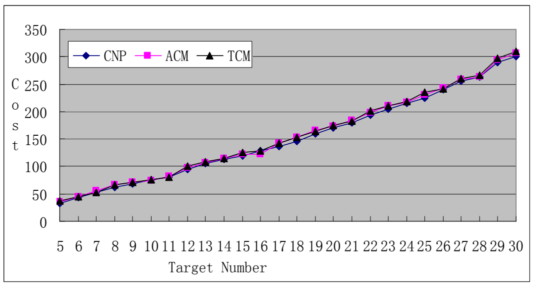


Figure 5. Comparison of task solving cost

From figure 5, the task solving costs of the three models are nearly equal, i.e. the advantage of the model mainly exhibits on the negotiation cost, and the task solving cost doesn't change obviously.

Experiment 2: Changeable Task Attribute, i.e. dynamic invader category

Similar to the above experiment, change the task attributes at every sampling in 10 tests.

Figure 6 shows the comparison of negotiation cost of the three algorithms in this condition:

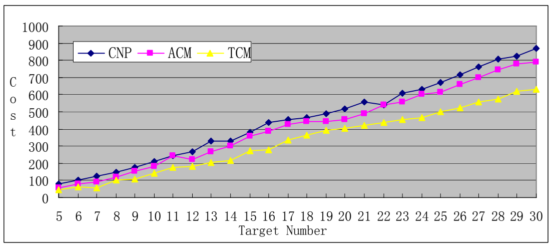


Figure6. Comparison of negotiation cost

From figure 6, when task attributes change frequently, the difference between the negotiation cost of ACM and CNM is not obvious, this is because the change of task attributes leads the descend of the coalition stability, and the negotiation cost of solving increases correspondingly. Acquaintance coalition has memory effect, so it has the advantage to the new tasks similar to the tasks it solved before. TCM has bigger advantage compared with ACM and CNM, though it shows ascend trend.

Figure 7 shows the comparison of task solving cost of the three algorithms:

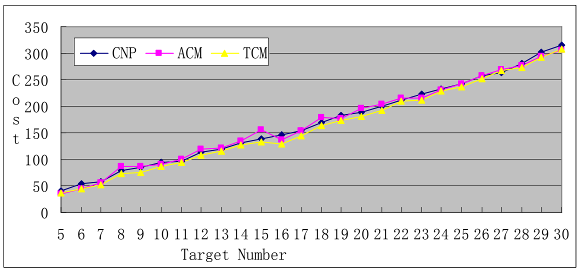


Figure 7. Comparison of task solving cost

From figure 7, the task solving cost of ACM exceeds CNM sometimes. This is because when coalition is unstable, the negotiation result can't be returned at the given time, which leads to the coalition acquiring the task is not the optimization result. The negotiation cost of TCM is nearly equal to that of CNM, which shows TCM can't solve the solving optimization problem effectively, and needs to introduce other methods.

Experiment 3: Comparison of Task Allocation Performance

The performance of ACM and TCM is compared from the numbers of seeking optimization solution and the numbers of plunging local optimization solution. 100 tests are carried on, and the number of tasks is generated stochastically in every test, as shown in table 1:

TABLE 1.

PERFORMANCE OF THE ALGORITHMS

	ACM	TCM
numbers of seeking optimization solution	70	81
numbers of plunging local optimization solution	30	19

From table 1, the task allocation algorithm of TCM is improved largely in searching capability than ACM. It improves the stability, avoids the premature convergence better, overcomes plunging local optimization better at task allocation, and has a higher probability to find the global optimization solution.

VI CONCLUSION

This paper presented a MAS negotiation model based on task coalition and combination algorithm of GA and simulate annealing, and defined task coalition and its generation, and maintenance strategy. A combination of GA and simulate annealing algorithm is proposed to optimize task allocation, which improves the efficiency of task allocation and reduces communication cost of sub-task allocation in the coalition. By a missile defense system, test and analyze the effect of the negotiation model. It is proved that the model can effectively reduce the negotiation cost and improve the efficiency of task allocation with the optimized genetic algorithm.

The model has some aspects needed to improve, they are: 1) The first formation of task coalition allocates tasks stochastically first, which leads the coalition organization relies on the first task execution greatly, and not favor the coalition stable in some situations and increase the extra cost; 2) When task characters change greatly, the stability of coalition needs to be studied further; 3) The execution of Agent join strategy may lead the distribution of system resources asymmetric, and need to add the factor of resources distribution in join strategy.

ACKNOWLEDGEMENTS

This work is supported by the Scientific Research Project of Zhejiang Education Department (Y201226267). Thanks for the help.

REFERENCES

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

[2] Larson, K., Sandholm, "Anytime Coalition Structure Generation: An Average Case Study", *Journal of Experimental and Theoretical AI*, vol.12, pp.40-47, January 2000.

[3] Conitzer, V., Sandholm, T, "Complexity of Constructing Solutions in the Core Based on Synergies among Coalitions", *Artificial Intelligence*, vol.170, pp.607-619, February 2006.

[4] Stefan Warwas, Klaus Fischer, Matthias Klusch, Philipp Slusallek, "BOCHICA: A Model-driven Framework for Engineering Multiagent Systems", *In Proc. of the 4th Int. Conf. on Agents and Artificial Intelligence (ICAART)*, pp.109-118, February 2012.

[5] Eric Sodomka, Eric Sodomka, John Collins, John Collins, Maria Gini, Maria Gini, "Efficient statistical methods for evaluating trading agent performance", *Proc. of the Twenty-Second National Conference on Artificial Intelligence*, pp.770-775, July 2007.

[6] Umesh Deshpande, Arobinda Gupta, and Anupam Basu, "Performance Enhancement of a Contract Net Protocol Based System Through Instance-Based Learning", *IEEE Transactions on Systems, Man and Cybernetics—Part B: Cybernetics*, vol. 35, no. 2, pp.345-358, April 2005.

[7] Luo Jianbin, Hu Shangli, Lin Yaohai, "Coalition structure generation with given required bound based on coalition combination", *3rd International Conference on Intelligent System and Knowledge Engineering*, pp.555-559, November 2008.

[8] T. Rahwan, S. D. Ramchurn, N. R. Jennings and A. Giovannucci, "An Anytime Algorithm for Optimal Coalition Structure Generation", *Journal of Artificial Intelligence Research (JAIR)*, vol. 34, pp.521-567, March 2009.

[9] Zhang Xinliang, Shi Chunyi, "A Dynamic Formation Algorithm of Multi-Agent Coalition Structure", *Journal of Software*, 18(3), pp.574-581, March 2007.

[10] Tomasz Michalak, Andrew Dowell, "Pre-processing techniques for anytime coalition structure generation algorithms", *Springer LCNS*, vol.5605, pp.99-113, 2008.

[11] Xiaowei Zhang, Bin Li, Junwu Zhu, Jun Wu, "Utility Based Optimal Task Scheduling Problem in a Multi-agent System", *JDCTA: International Journal of Digital Content Technology and its Applications*, vol. 4, no. 9, pp.27-35, January 2010.

[12] Jiang Jianguo, Xia Na, Qi Meibin, Mu Chunmei, "An ant colony algorithm basd multi-task coalition serial generation algorithm", *Acta Electronica Sinica*, vol.33, no.12, pp.2178-2182, December 2005.

[13] Yin Xiang, Jiang Jianguo, Xia Na, "Multi-task Multi-coalition Generation Problem: Model and Algorithm", *Systems Engineering -Theory & Practice*, vol. 28, issue 4, pp.90-95, April 2008.

[14] Walaa H. El-Ashmawi, Hu Jun, LI Renfa, "A Novel Distributed Fuzzy-based Negotiation Model for Coalition Formation in Multi-Agent Systems", *IJACT: International Journal of Advancements in Computing Technology*, vol. 4, no. 15, pp.270 -279, September 2012.

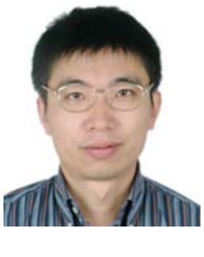
[15] Mukun Cao, "Decision Making Model for Intelligent Agent in Automated Negotiation", *Journal of Software,* Vol 6, No 8 (2011), 1537-1544, Aug 2011

[16] Bailing Liu, "Efficient Trust Negotiation based on Trust Evaluations and Adaptive Policies" *Journal of Computers*, vol 6, no 2, pp.240-245, Feb 2011.

[17] Tao Haijun, Wang Yadong, Guo Maozu, Wang Hanlun, "A Multi-Agent Negotiation Model Based on Acquaintance Coalition and Extended Contract Net Protocol", *Journal of Computer Research and Development*, vol.43, no.7, pp.1155~1161, July 2006.

[18] Hai-jun Tao, Ya-dong Wang, Mao-zu Guo. "An Extended Contract-Net Negotiation Model Based on Task Coalition and Genetic Algorithm", *Proceedings of 2007 International Conference on Machine Learning and Cybernetics*, Hongkong, vol 2, pp.879-884, July 2007.

Xiang Lin, born in 1975, is a lecturer in China Jiliang University. She received master degree from Harbin Institute of Technology, and her major is computer application. Her research interests are in the areas of computer architecture and multi-agent system theory.



Haijun Tao, born in 1975, is Ph.D. and associate professor in China Jiliang University. He received PhD degree from Harbin Institute of Technology, and his major is computer application. His research interests include multi-agent system theory and machine learning technology.