

## Using Counter-Proposal to Optimize Genetic Algorithm Based Cooperative Negotiation

Liu Ping, Hu Yongtong

Yan Yuhong, Zheng Danian, Yang Shiyuan

Automation Department

Tsinghua University

Beijing, 100084, P.R.China

**Abstract**—Multiagent systems have been widely used in concurrent engineering. However, most multi-agent tasks have high complexity. Optimal solutions of them are nearly intractable due to the heavy load of communication and large complexity of knowledge exchanges. In this paper, we extend the agent structure in our previous work[1]. Rather than using common multi-agent diagram, we use stochastic search techniques such as genetic algorithms (GAs) to explore the solutions. Moreover, we also add counter proposals to speed up the searching process. Experimental results about designing a gear pump are satisfying, which turn out that the combination of GAs (modified to adapt to counter proposals) and Multiagent Systems can considerably speed up the convergence process, and at the same time, guarantee the best globe solution.

### I. MOTIVATION

This paper is concerned with the development of a concurrent engineering environment that supports design products upon customs' requests. In this environment, product life-cycle elements, such as design, manufacturing, transportation, logistics etc. are taken into consideration simultaneously even in early design phase. Multi-agent system, because of its implicit parallelism and distributed nature, was adopted as the means of coping with these kind of industrial applications. These problem-solving agents, through communication and negotiation, "are not trying to enforce their views on one another or to maximize their own benefits, but rather are trying to share their individual knowledge and beliefs in order to determine what really is best." [2] Therefore, they can achieve lower cost, improve the product quality and then, secure marketing shares.

Many researchers have done lots of work both on theories and applications. However, during the research on multi-agent system, it was found that one of the most time-consuming and difficult activities when building multi-agent systems was ensuring that the community acted in a coherent manner even when unplanned events occurred. [3] And most multi-agent tasks have high complexity. Optimal solutions of them are nearly intractable because of the heavy communication load and large negotiation complexity. However, "Negotiation is search", [4]. Negotiation process can be viewed as negotiators jointly searching a multi-dimensional space and then agreeing to a single point in the space. In the negotiation process, each

dimension corresponds to an attribute to be negotiated, and each attribute has two or more alternatives that are indexed by elements of a set. So it is possible for us to use mathematical tools to model the process. GA is just suitable for this purpose. With the help of GA, we can represent attributes as 'genes', thus, take the advantage of GAs to speed the negotiation process searching for globally optimized proposal by turning the normally negotiation process into a stochastic searching process for maximum/minimum-point problem. While in multiagent systems, counter-proposal is an important feature, and counter-proposal, when expressed in mathematical version, can be viewed as searching direction, thus, the combination of GA and counter proposals can be expected to speed up the GA's convergence speed.

### II. GENETIC ALGORITHM IN MULTIAGENT SYSTEM

Genetic Algorithms have many important features that make them very suitable to work in multiagent systems:

- Genetic algorithm is based upon Darwinian evolution. It has implicit parallelism and genetic mechanism, theoretically, it can be applied to concurrent engineering.
- Genetic algorithms have open and flexible structure, and there is no need for agents to exchange domain knowledge between each other. Therefore, GAs can be easily modified to adapt to multiagent systems.
- Genetic algorithms search for global optimal solution. And the searching result has nothing to do with initial values. Through passing the fittest chromosomes to the next generation, modified GAs can guarantee the global optimal solution.[5]

In GAs, candidate solutions to the problem are encoded into "chromosomes", which are representations of a solution or instance of the problem at hand. Each chromosome is assigned a 'fitness' value according to their performances in the evolution process. In the context of negotiation in this paper, each of the software agents begins with a population of various, randomly generated (and not necessarily good) negotiation proposals. We employ three genetic operators to create new proposals: reproduction, crossover, and mutation with counter proposals.

Reproduction uses chosen "parents" and combines them to create new candidate solutions that comprise the next generation.

Cross Over swaps sub chromosomes from two parents. Here we select single point crossover approach.

Normal Mutation transforms the proposal by changing chromosome randomly without keeping the original. In this paper, besides alter the "genes" randomly, we also mutate genes by counter proposals.

A proper fitness function is also essential to the success of GAs, which will be used to evaluate each "chromosome" and then determine which one will be passed into the next generation.

### III. FORMULATE THE NEGOTIATION

In order to make agents understand each other in genetic algorithms, first we must code concerned characteristics to be applicable for GA (section 3.1). We also need a value function to evaluate each possible solution and then assign the evaluation as its 'fitness', which will determine whether it will live on in next generation or just is discarded. In this paper, we will discuss a designing process of a gear pump. We will explain the process with the help of the example.

#### A. Encode

Using Vector  $X = [x_1, x_2, \dots, x_n]$  to represent basic characteristics of a gear pump, for example,  $x_1$  represents material,  $x_2$  represents turning speed. Then  $X$  (chromosome) represents a point in the whole solution space. Some attributes, such as geometry size, are coded continuously. While others, such as material attribute, are discretely coded.

#### B. Value Function

In general, the value,  $V$ , of a particular option  $X = [x_1, x_2, \dots, x_n]$  of each agent is

$$v(X) = f(x_1, x_2, \dots, x_n)$$

Where  $V_i$  is the evaluation function for the alternatives for gene  $i$ , then, the globe evaluation function is

$$V = f(v_i(X))$$

In our model, we concern most about issues of cost, the lower, the better. Value function can be described as below:

$$Fitness_i = 1 / \sum_j COST_{j,i} \quad (1)$$

Where  $i$  means the order of chromosome,  $j$  represents the order of agent.

#### C. Counter-proposal

Evolutionary process deals with populations of individuals. In our case, we consider populations of proposals, described as below:

$$\langle proposal \rangle = \langle y_1, y_2, \dots, y_n \rangle$$

and,

$$\langle counter-proposal \rangle = \langle y_1, y_2, y_3, \dots, y_n \rangle$$

is used to model agents' reaction (counter-proposal) to a proposal.

Agents in the negotiation can influence the searching direction by contributing their opinions through counter proposals. How will a counter-proposal be generated? We know that each agent tends to evaluate the proposal from its local view, we can't expect it alone to have sufficient knowledge to choose globally optimized proposal. Thus, conflicts between agents arise inevitably. When an agent disagrees with one proposal, it can modify the related "genes" locally, for example, design agent may change the material  $y_1 = \#45$  Steel (material No.1) to material

$y_1 =$  Alloyed Steel (material No.2), while keeping other attributes unchanged. The newly formed count-proposal will be added to chromosome population, and take part in next cycle of computation. The major advantages of counter proposal is that it can help to quickly find the feasible solution space, then in the feasible solution space, search the optimal resolution. Thus, the searching process can be greatly speeded up.

#### D. Genetic Computation

##### 1) System Structure

Here we use a manager agent to organize the GA computation. Its responsibilities include:

- collect all evaluations made by other agent;
- assign "fitness" value to each "chromosome" based on the fitness function;
- execute genetic computation.

Each agent will evaluate the proposals according to its local interests. Then, prompt counter-proposals when they find out that their local interests are not optimized.

##### 2) System Running Pseudo Code

The genetic computation process can be described as following:

Procedure GA program supporting negotiation

Begin

$t := 0;$

Initialize first generation  $P(0);$

While not (specified stop condition) do

Begin

Agents Prompt Counter-proposals;

Insert new chromosomes based on counter-proposals;

Evaluate(  $P(t)$  );

```

Reproduce( P(t) );
Crossover( P(t) );
Mutate( P(t) );
t:= t+1;
End;
End;

```

#### IV. DEMONSTRATION

A multiagent system is to design gear pumps. There are 6 features to describe a gear pump: volume, throughput, output pressure, turning speed of the gear, Z and m of the gear, thermal treating, bearing and material

The system consists of 5 agents: Negotiation Manager agent, Sales agent, Design agent, Manufacturing agent, and Transportation agent. Detail information of the structure of this system please refers to [1].

The functions of each agent is described as following:

The Negotiation Manager Agent controls the process of negotiation based on GAs. In each round of negotiation, this agent is in charge of genetic computations. This agent evaluates each plan by calculating the cost by adding up the cost of each agent. Some GA parameters adopted in the Negotiation Manager Agent are shown as following:

TABLE 1 SYSTEM PARAMETERS

Population size	Total generation	Pc	Pm
20	200	0.8	0.1

The Sales Agent checks whether a plan satisfies custom requirements. Here the custom requirements is that the throughput should be higher than 30. If a plan does not match, the Sales Agent will report an infinitive high "cost of sales" (100000 YUAN here), and present a counter proposal with higher throughput.

The Design Agent calculates the cost of design according to throughput and turning speed of the gear. For example, if the throughput is among a standard series, say 8, 12, 18, 29, 38, 60 or 120, the cost of design will be low (500 YUAN here), because the design reference book can be of help. Otherwise, if the throughput is between 1 and 120, but not a standard one, the cost of design will be higher (2000 YUAN here), because the plan is a special one. In these cases, the Design Agent will also present counter proposals to adopt standard throughput. If the throughput is larger than 120, the cost of design will be very high (5000 YUAN here), because a totally new gear pump has to be designed.

The Design Agent also checks the validation of some features. For example, it calculates the permitted throughput  $T_p$  according to a formula

$T_p = f(\text{turning-speed, output-pressure, } Z, m)$  If the throughput  $T$  in a plan is larger than  $T_p$ , which means the plan is infeasible, the Design Agent will report an infinitive high "cost of design" (100000 YUAN here), and prompts a

proposal with larger Z or m.

The Manufacturing Agent can calculate the cost of manufacturing according to the output pressure, Z and m.

The Transportation Agent cares only the volume of the gear pump, because larger geometry size of a gear pump means higher transportation fee.

Following are some result and analysis:

Figure 1 shows the evolution process of cost (without counter proposal). Although the cost of some agent rises,

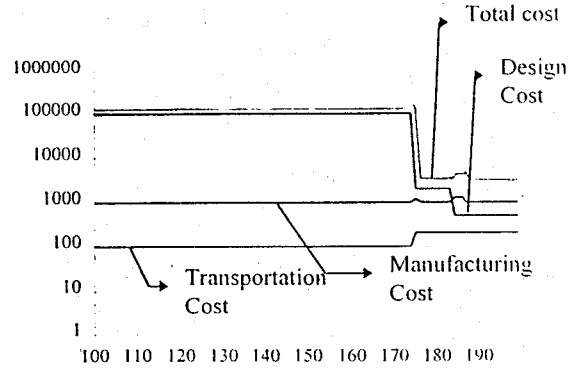


Fig.1 Evolution process of Cost (without counter proposal)

the total cost keeps on dropping. Tab. 2 shows some related detail information of the best scheme and the evaluations of each agents. From generation 1 to 174, the total cost stays at a high level. That is because the Design agent finds that the volume is too small to contain a gear with  $Z=20$  and  $m=11$ . In generation 175, a new plan with larger volume appears, then the cost of design drops significantly. In generation 184, the throughput rises to 38. This is a standard throughput. Although the manufacturing cost rises, the cost of design drops more. So the total cost drops. From generation 185 to 190, the cost of manufacturing keeps decreasing. This process can be viewed as searching optimal point in a limited area. In generation 190, the global optimal point is reached.

The convergence speed showed in Fig. 1 and Tab. 2 is slow. The reason is that agents just wait for a plan. Comparing with Tab. 2, Tab. 3 shows some detail evolution information about GA with "counter proposal" mechanism. And Fig. 2 shows related evolution process.

In the best scheme of the first generation, the volume is too small to contain a gear with  $Z=16$  and  $m=8$ . However, it took only 20 generations to solve the conflict, because first the Designer Agent suggested a scheme with smaller Z and m, then the Manufacturing Agent prompted a counter proposal with larger volume. After only 22 more generations, the multiagent system reached one optimal scheme. It was much faster than the GA based negotiation process without counter proposal mechanism. It's the "counter proposal" mechanism who make agents more active. In Fig. 3, the convergence speed of total costs are

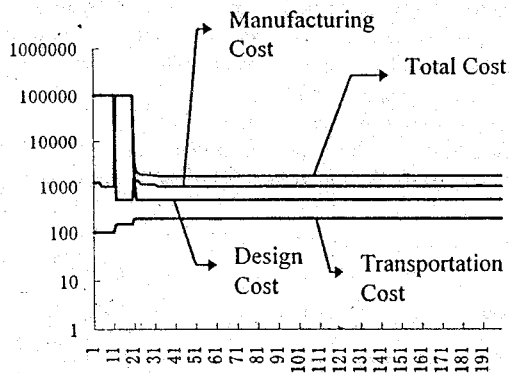


Fig.2 Evolution process of cost (with counter proposal)

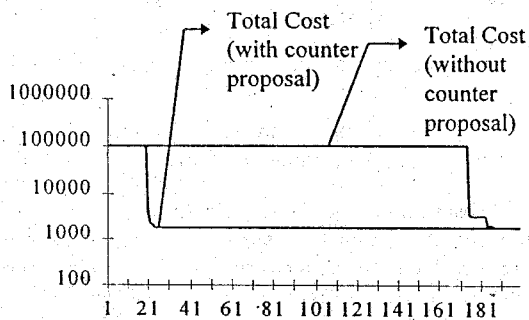


Fig.3 Comparing cost convergence speed of two methods

compared in Fig.3 to illustrate how this mechanism improves convergence speed significantly.

## V. CONCLUSION

In this paper, we prompt a novel way to organize cooperative negotiation process in multiagent process. We not only use genetic algorithm to formulate multiagent negotiation process, but also adopt counter proposal to speed up the convergence process of searching global optimal solution. Experimental results about designing a gear pump demonstrate that the combination of GA and agents' counter proposals would be an effective way to reduce communication load and complexity of knowledge exchanges in multiagent systems.

## VI. REFERENCES

- [1] "A Multiagent System For the Support of Concurrent Engineering". Hu Yongtong, Liu Ping, Yan Yuhong, Zheng Daniao, Ma Changchao, Jurgen Bode, Ren Shouju, P959-964, Proceedings of 96'IEEE Conference on System, Man, and Cybernetics.
- [2] "Electronic Catalog and Negotiations" CITM Working Paper 96-WP-1016, Carrie Beam and Arie Segev, Fisher Center of Information Technology & Management, Walter A. Hass School of Business, U. of California, Berkeley.
- [3] "Controlling cooperative problem solving in industrial multi-agent systems using joint intentions", N. R. Jennings, Artificial Intelligence 75 (1995) 195-240
- [4] "A machine Learning Approach to Automated Negotiation and Prospects for Electronic Commerce", Oliver, Jim R. <http://opim.wharton.upenn.edu/~oliver27/papers/jmis.ps>, July 31, 1996
- [5] "An overview of Genetic Algorithms", Xi Yugeng, Chai, Tianyou, Yun Weiming, Vol.13, No.6, Dec., 1996, Control Theory and Applications ( In Chinese)
- [6] Martin, S., Szapiro, T., Haigh, K., "Genetic Algorithms Approach to a Negotiation Support System", IEEE Trans. Systems, Man, and Cybernetics, Jan./Feb., Vol. 21, No. 1, P102-114

**TABLE 2: SOME DETAIL INFORMATION OF THE BEST SCHEME AND EACH AGENT'S EVALUATIONS (NO COUNTER-PROPOSAL)**

generation n	cost				best scheme in each generation								
	total cost	cost of design	cost of manuf.	cost of trans.	Volume	Throug hput	Turnin g speed	Output pressure	Z	M	Thermal treating	Bearin g	Mater- ial
1	101210	100000	1110	100	0.0347	53	1000	0.57	20	11	1	0	0
...													
173	101100	100000	1000	100	0.0381	53	699	0.45	20	11	0	0	0
174	101100	100000	1000	100	0.0381	53	699	0.45	20	11	0	0	0
175	3412	2000	1212	200	0.175	31	1377	0.45	20	8	1	0	1
176	3199	2000	1000	200	0.175	31	1377	0.45	20	8	0	0	0
...													
182	3199	2000	1000	200	0.175	31	1377	0.45	20	8	0	0	0
183	3199	2000	1000	200	0.175	31	1377	0.45	20	8	0	0	0
184	1972	500	1272	200	0.1864	38	1377	0.45	20	8	1	1	1
185	1972	500	1272	200	0.1864	38	1377	0.45	20	8	1	1	1
186	1972	500	1272	200	0.1864	38	1377	0.45	20	8	1	1	1
187	1709	500	1010	200	0.175	38	1377	0.45	20	8	1	0	0
188	1709	500	1010	200	0.175	38	1377	0.45	20	8	1	0	0
189	1709	500	1010	200	0.175	38	1377	0.45	20	8	1	0	0
190	1700	500	1000	200	0.1864	38	1377	0.38	20	8	0	0	0

**TABLE 3: SOME DETAIL INFORMATION OF THE BEST SCHEME AND EACH AGENT'S EVALUATIONS (WITH COUNTER PROPOSAL).**

Ge ner atio n	total cost	cost of design	cost of manuf.	cost of trans	Volum e	Throug hput	Turnin g Speed	Output pressur e	Z	m	Ther -mal treat	Bearin g	Mate -rial
1	101299	100000	1200	100	0.0354	29	1273	0.42	16	8	0	0	1
11	101100	100000	1000	100	0.0354	29	1273	0.42	16	11	0	0	0
12	100650	500	100000	150	0.0673	38	1273	0.29	1	13	0	0	0
...	...	...											
20	100650	500	100000	150	0.0673	38	1273	0.29	1	13	0	0	0
21	3531	2000	1332	200	0.1368	50	1497	0.58	17	7	1	0	1
22	2032	500	1332	200	0.1368	38	1497	0.58	17	7	1	0	1
23	2032	500	1332	200	0.1368	38	1497	0.58	17	7	1	0	1
24	1809	500	1110	200	0.1368	38	1497	0.58	17	7	1	0	0
...	...	...											
28	1800	500	1100	200	0.1368	38	1497	0.58	17	7	0	0	0
...	...												
32	1709	500	1010	200	0.1368	38	1497	0.48	17	7	1	0	0
33	1700	500	1000	200	0.1368	38	1497	0.48	17	7	0	0	0