A Multi-Agent Approach for Solving Optimization Problems involving Expensive Resources

Hoong Chuin Lau
School of Computing
National University of Singapore
lauhc@comp.nus.edu.sg

Hui Wang
Industrial and Systems Engineering Dept.
University of Southern California
wang.hui@usc.edu

ABSTRACT

In this paper, we propose a multi-agent approach for solving a class of optimization problems involving expensive resources, where monolithic local search schemes perform miserably. More specifically, we study the class of bin-packing problems. Under our proposed Fine-Grained Agent System scheme, rational agents work both collaboratively and selfishly based on local search and mimic physics-motivated systems. We apply our approach to a generalization of bin-packing - the Inventory Routing Problem with Time Windows - which is an important logistics problem, and demonstrate the efficiency and effectiveness of our approach.

Categories and Subject Descriptors

I.2.11 [Distributed Artificial Intelligence] *Coherence and coordination, Intelligent agents, Multiagent systems*

Keywords

Multi-agent system, optimization problems, expensive resources.

1. INTRODUCTION

Among optimization techniques, agent-based approaches distinguish themselves by their strength in modeling the problem's distributed nature and expressing complex relationships among entities within the system. In agents literature, resource allocation problems have been modeled and solved as Distributed Constraint Satisfaction Problem (DCSP) [1] and Distributed Constraint Optimization Problem (DCOP) [2]. Unfortunately, these approaches tend to produce weak results when extended to solve optimization problems that have rich combinatorial structures such as the bin-packing problem. One reason is perhaps the underlying tacit representation of one variable per agent. One method for handling multiple local variables per agent was proposed in [3], where each agent first finds all solutions to its local problem. By obtaining all solutions to local problems, the given problem can then be re-formalized as a DCSP, in which each agent has one local variable whose domain is a set of obtained local solutions. In doing so, agents can apply the algorithms that worked on single local variables. One drawback of this method is that when a local problem becomes large and complex, finding all the solutions of a local problem becomes virtually impossible.

A key limitation in the above mentioned works is that they did not consider the issue of resource optimization. The goal of DCOP is to minimize the number of conflicting constraints rather than the cost of resources utilized. Many real-world resource optimization problems can be seen as generalizations of the bin-packing problem, where each resource is capacitated and the goal is to satisfy a given set of tasks using the minimum resources (bins). In many cases, each resource is expensive and hence the power of an algorithm lies in its ability to reduce the objective value by even a very small value.

Much resource allocation research from the AI community makes use of local search due to its computational efficiency. However, local search alone may be ineffective on problems with where the number of resources is small (scarce), and the goal is to minimize that. A classical example is the Vehicle Routing Problem with Time Windows where the primary objective function is to minimize the number of vehicles. In this problem, local search performs badly – because it is not easy (or even possible) to predict whether the number of vehicles can be reduced with successive local moves in the long term.

This paper looks at a computationally efficient multi-agent scheme for the above-mentioned problem. More specifically, we propose our Fine-Grained Agent System (FGAS) for solving a class of bin-packing problems with the following basic features:

(1) the problem is to insert items into bins with fixed capacity;

(2) the primary objective of the problem is to minimize the number of bins.

Obviously, vehicle routing problems fall into this category where the routes can be modeled as bins while customers as items. Obviously also in this case, the sequence of items within the bins is important since it involves another objective of minimizing the total travel distance.

What distinguishes our approach from previous multi-agent approaches for optimization is that the FGAS agents are endowed with more intelligence and stronger human behavior. The problem's objectives are expressed through the local objectives of agents. In other words, each agent has its own objective, which needs to be consistent with the objectives of the problem. Agent actions are motivated by their objectives. A novelty of FGAS is how the agents interpret the objective of minimizing the number of bins within its local view. Another novelty of FGAS lies in how FGAS evaluates a move - besides the change in the overall objective value, the evaluation also includes a *long-term effect* or the *future expectation* of a move, which is computed by agents based on its local information, as well as information provided by *constraint agents* on the degree of satisfying or violating a constraint.

Since FGAS has different types of agents, to minimize confusion, we use the term *subagent* henceforth to refer to the agent in a standard multi-agent system.

Due to the limitation of the local knowledge, subagents may hold different opinions towards the best move, and this phenomenon is termed as *conflict* in this paper. Conflicts can be resolved via negotiation amongst agents. An example of cooperativeness-based negotiation strategy was introduced in [4]. In their work, the conflict is defined as the constraint violation caused by some agents. When conflict occurs, agents will resolve it by selecting values not only to satisfy the constraint but also maximize the other agents' flexibility of choosing values. In this way, they achieve a potentially fast solution convergence. FGAS extends their idea by estimating each move's future expectation and quantifying it into a value. The one with best future expectation will be chosen as the next move. Experiments show that FGAS are capable of converging to good solutions quickly.

This paper proceeds as follows. In section 2, we present the FGAS model, followed by an in-depth discussion of the agents in section 3. In section 4, we apply FGAS to a practical problem in logistics and scheduling – Inventory Routing Problem with Time Windows – which is a generalization of bin-packing, where vehicles and warehouses are seen as bins. We conclude in section 5.

2. FGAS Model

The FGAS model is extended from the multi-agent optimization system proposed by Tsui and Liu [5]. We define FGAS as a 4-tuple <*AA*, *CA*, *SA*, *L*>, where *AA* is an arbitration agent, *CA* is a set of constraint agents, *SA* is a set of subagents $SA = \{sA_1, sA_2, \ldots sA_n\}$, *L* is the local behavior of agents.

2.1 Arbitration Agent *AA*

AA is a virtual agent in FGAS that serves as a global information provider for the subagents. It behaves like the environment ε defined in Tsui and Liu's model ([5]).

Besides its role as a global environment information provider, *AA* also provides the necessary centralized concurrency control for a problem where certain kinds of constraints exist among subagents. For example, if the change of subagent *a* affects the local knowledge of subagent *b*, they cannot be computed concurrently. Otherwise, *b* will be acting based on obsolete information. Such agents are defined as *dependent subagents*. We can handle this case by using *AA* monitoring the system and preventing dependent subagent changing their states concurrently. In this sense, FGAS is able to solve more problems by employing *AA*.

2.2 Constraint Agents *CA*

We observe that the degree of violating or satisfying constraints is very helpful information for subagents to estimate a change's potential effect on the solution quality. In FGAS, we introduce a class of constraint agents to represent constraints. Constraint agents work as consultants for subagents, providing information to help subagents make decisions. As an example, recall that the primary objective of the bin-packing problem is to minimize the number of bins without exceeding the bin's capacity. Consider the scenario where the bin capacity is 200 and its load after accepting some change becomes 5. This change will probably be accepted, since it gives a good future expectation of reducing one bin. This information of the bin's load upon accepting a change is sent to the subagents, when the associated constraint agents check the degree of violation of the vehicle capacity constraints.

2.3 Subagents *SA*

Subagents form the backbone of FGAS. Each subagent represents an independent entity in the system. In bin packing for example, a subagent represents a "bin" with a prescribed capacity. To solve an optimization problem, subagents adopt the local search approach: a list of moves will be proposed when several subagents encounter with each other. The outcome of a move may change subagents' internal state and improve the objective value. When a subagent evaluates a move, the *AA* and corresponding *CA*s will be consulted for information. Through agent communication, negotiation and cooperation, subagents collaborate to optimize the problem objective.

Subagents exhibit the following characteristics:

a. Cooperative

Subagents are generally cooperative in the sense that if a change can lead to the improvement of the overall objective value, it will be desired by all subagents involved. FGAS subagents are not always cooperative in the sense that they will intelligently form their own opinion of an action's future expectation. They will selfishly stick to their own opinion by rejecting an action that can improve current objective value or accepting an action that can degrade current objective value.

b. Selfish and Intelligent

Subagents are intelligent in the sense that they can compute a move's future expectation based on both 1) their local information and 2) the information from a constraint agent. If a move has good future expectation, the subagent will have strong desire to accept it by sacrificing the current objective value. On the contrary, if a move has an undesirable future expectation, the subagent will selfishly reject it. The selfishness and intelligence of subagents give us a fast solution convergence speed, as we can see from the experiment results.

c. Negotiable

When subagents involved in one move have different opinions towards it, a conflict is said to occur and negotiation kicks in. By negotiation, we mean each subagent is willing to make compromise to give up its own opinion. This can be seen as another form of cooperativeness among subagents.

2.4 Agent Coordination

We now present the overall agent coordination. Our idea is inspired from natural systems in which complex global structures and behaviors result from local interactions among many simple elements. In FGAS, we develop a physics-motivated coordination mechanism [6] that treats subagents as randomly moving, locally interacting entities. A similar scheme has already successfully used in the area of coalition formation for large-scale electronic markets in [11]. Note that in E-markets however, the agents can randomly move and interact since they are not constrained by constraint agents. In our scenario, there may exist external constraints that limit concurrent computation. For example, if the change of subagent *a* affects the local knowledge of subagent *b*, then they cannot be computed concurrently. Otherwise, subagent *b* may make wrong decision based on the obsolete information. The *AA* is used to prevent the situation of error arising from concurrent computation with inter-dependent agents.

The initialization of *AA* and *CA* are straight-forward. A standard algorithm, such as a greedy algorithm, is used to generate an initial solution. After initialization, subagents begin the process of collaboration and negotiation. A subagent will interact with another on a random basis following the physics-motivated coordination mechanism of [6]. *AA* will be operating in the background, ensuring the correctness of concurrent computation. When subagents encounter each other, a move list will be proposed based on the set of predefined moves. The moves will be evaluated by each subagent based on the information both locally available and obtained from the *CA*s. When a conflict occurs, subagents begin negotiation with one another. The move with best future expectation will be applied. The overall objective value is recorded by *AA*. When a termination criterion is met, the program ends.

3. Subagents

As discussed in the previous section, subagents are the most important kind of agents in FGAS. In this section, we provide further technical details. An FGAS subagent i is defined by a 6-tuple A_i=<Sta_i, RC_i, O_i, X_i, $D_i(m)$, $P_i(m)$>, where Sta_i is the state of subagent *i*, RC_i is the set of constraint agents related to subagent *i*, O_i is the set of objectives of subagent, X_i is the variables associated with the subagent, $D_i(m)$ measures how the subagent desires a move *m*, and $P_i(m)$ is the negotiation component of a subagent.

3.1 State

We introduce a state for each subagent, defined as follows:

if Sta_i = active, sA_i is able to change its state

if Sta_i = non-active, sA_i is unable to change its state.

The responsibility for checking and informing the states of inter-dependent subagents rests with the *AA* so as to ensure the correctness of the system and at the same time maximize possible concurrent computation without violating those constraints.

3.2 Objectives

Although the objective of a subagent is coherent with the global objective, subagent may only see and direct its action based on its local objective. In terms of bin-packing, a naive local objective function is the objective of destroying itself by pushing out all the items in a bin to other subagents. However, such approach is impractical because when a bin (say *a*) transfers its items to others (say *b*), the load of *b* will increase, which violates *b*'s objective and result in an oscillating solution improvement process. Hence, each subagent's local objective of minimizing number of bins is defined as Figure 1:

1) when a subagent detects that it stands a chance of emptying all its items, it will encourage the change that transfers its items to others;

2) when a route is almost full, the subagent has the desire to top up its load, but without violating capacity constraint.

The second part of the local objective is based on the observation that when the total demand is fixed, the fewer bins are used, the more items one bin will have to contain.

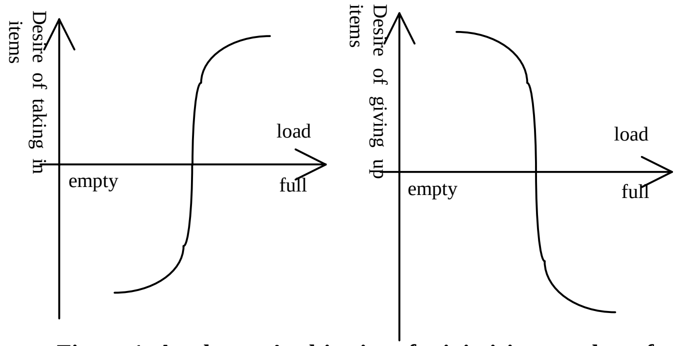

Figure 1. A subagent's objective of minimizing number of bins

3.3 Desire

The desire of subagents quantifies how much a subagent desires a move according to its objective. Formally, a desire value is defined as:

$$D_i(m)= \alpha_0 * u_m + \alpha_1 * e_i(X_i,C_i)$$

where *i* is the index of the subagent; *m* is the move; $\alpha_0 + \alpha_1 = 1$; u_m is the overall change in objective value by applying the move m; $e_i(X_i,C_i)$ gives the future expectation function which is defined on two variables: X_i being the local information, and C_i being the information obtained from the related constraint agents.

Hence, a desire value consists of 2 parts:

1.*Changes in the overall objective value.* It represents the cooperative aspect of a subagent. The more a move contributes to the overall objective, the more a subagent desires the move.

2.*Future expectation.* This contains the information of future expectation, which represents the selfishness and intelligence of a subagent, derived from not only from local information but also information from its related constraint agents.

3.4 Negotiation

When a list of moves is proposed, each subagent involved will evaluate the moves in the list and select the best one. When they have different opinions towards the best move, a conflict occurs. In FGAS, agent negotiation is triggered to deal with the conflict between subagents. By negotiation, we mean subagents will compromise by giving up its own preference defined according to the move's contribution to overall objective and its future expectation.

Mathematically, the best moves from each subagent will be re-evaluated using the function $p(m)=P(u_m,e_m)$, where u_m is the immediate overall objective value change by applying the move *m*, and e_m is the sum of the future expectation from each subagent involved. The one with highest $p(m)$ value will be chosen. This negotiation strategy can be seen as an extension from cooperativeness-based strategies of [4] in the sense that a value that maximizes the chance of getting a good solution stands a higher chance to be chosen.

4. Application

We now apply FGAS to solve *Inventory Routing Problem with Time Windows* (IRPTW), which is a generalization of bin-packing, and more precisely, a generalization the well-known Vehicle Routing Problem with Time Window. IRPTW has important applications in logistics and transportation.

4.1 Problem Definition

Formally, IRPTW is defined as given:

C: set of customers;
T: consecutive days in the planning period $\{1,2,\ldots,n\}$;
d_{it}: demand of customer i on day t;
q_v: vehicle capacity;
q_w: storage capacity of customer i;
W_{it}: time window of customer i on day t;
c_{ih}: inventory cost per unit item per day at customer i;
c_{ib}: backlog cost per unit item per day at customer

output the following:

1. a distribution plan, i.e. flow amount from the warehouse to each customer i on day t; and
2. a set of daily transportation routes which carry the flow amounts in 1. from the warehouse to the customers

such that the sum of the following linear costs is minimized: (a) inventory cost at the warehouse and the customers, (b) backlog cost, and (c) transportation cost from the warehouse to the customers.

The distribution plan must obey the demands and storage capacity constraints, and the transportation routes must obey the standard routing, vehicle capacities and time windows constraints.

A solution instance S of IRPTW is viewed as a set of routes, each serving a certain subset of customers at a sequence of particular time points, i.e. $S = \{R_1, R_2, \ldots R_n\}$, where $R_i=\{(C_{i1},q_{i1},t_{i1}), (C_{i2},q_{i2},t_{i2}), \ldots, (C_{in}, q_{in}, t_{in})\}$, q_{ik} and t_{ik} are the supply amount and service time to customer k.

4.2 Modeling IRPTW using FGAS

Each subagent represents a route. In [7], it has been shown that the above problem can be seen as a multi-objective optimization problem that minimizes the transportation cost, inventory cost, and the number of vehicles. Within a subagent's local view, the objectives of minimizing transportation and inventory costs remain identical globally, while that of minimizing number of vehicles is translated as follows: the lower the load a route currently carries, the more it desires to empty its load; the higher the load a route carries, the more it desires to take a higher load. Hence, the future expectation function is defined in terms of how near the resulting load is from the two extreme values (0 and full capacity). As an example, suppose the full capacity is 1000, and a route with an existing load of 400 is evaluating a move that deletes from it a customer with demand of 300. The vehicle capacity constraint agent, besides checking that the route is still valid after the move, also sends to the subagent the remaining load after applying the move, which is 100. Since 100 is near to the extreme value 0 with respect to the range [0,1000], the future expectation value associated with this move will be high, according to the function shown in Figure 1.

The computation of inventory cost requires the information of all customers whose inventory levels are affected. This information is provided by the *AA*. Concurrent computation is limited by the inventory cost as follows. If two subagents serve the same customer in different days, they are dependent agents and cannot be active simultaneously. By checking the state component of each subagent involved, *AA* provides concurrency control.

We create 3 types of constraint agents that respectively handle the time windows, vehicle capacity, and customer holding capacity constraints. We introduce 2 local moves: the movement of items between agents *within* the same day (which are the classical *swap*, *relocate* and *exchange* moves for VRPTW, like [8]); and the movement of items between subagents *across* days, which follows the *transfer* move defined in [9].

4.3 Experiment Settings and Results

Following the test instance generation strategy of [7], our test cases are generated based on Solomon test cases for VRPTW. The planning period is 10 days. The vehicle capacity, locations and time-windows of the customers and depot are those specified in the Solomon instances. The demand d_{it} of customer i for day t is equal to the demand d_i of the Solomon instance, by partitioning the value $10*d_i$ into 10 parts, i.e. $d_{i1}, d_{i2},\ldots,d_{i,10}$ randomly such that d_{it} is within the range $[0.5*d_i, 1.5*d_j]$. The capacities of consumers and warehouse are the vehicle capacity and infinity respectively. As for cost coefficients, the inventory cost and backlog cost for each customer are 1 and 2 respectively. The transportation cost of each route is 10 times its total distance.

We conduct experiments on a PIII 1.13GHz machine. In Tables 1 and 2, we respectively compare solution quality and time performances between the solutions of HASTS [10] and FGAS. The D and V columns stand for total distance traveled and number of vehicles used, respectively. The column $T_{H=F}$ refers to the time needed for HASTS to reach the solution whose distance traveled is as good as FGAS.

We observed that FGAS effectively uses less vehicles compared with HASTS, especially for R and RC cases. However, the distance traveled is not as good as HASTS (average gap is 6.1% and the worst case is R206, by 10%). In terms of run time, FGAS has a much faster solution convergence speed (8 times running time on average for HASTS to reach the solution with the same quality as FGAS). We also conduct experiments to demonstrate the important contribution of agent selfishness and intelligence. Without selfishness and intelligence, note that both the total distance and number of vehicle is not good, which can be seen clearly by comparing the columns D_F vs $D_{Fw/oI}$, and V_F vs $V_{Fw/oI}$ in Table 1.

5. Conclusion

We proposed FGAS for solving a class of bin-packing problems with the aim towards solving real-world logistics problems involving expensive resources such as vehicles and warehouses. FGAS is able to overcome the shortcoming of local search by predicting the long-term effects of local moves through the computation of desire values in the context of multi-agent coordination and negotiation. Experimental results show that FGAS produces good solutions to IRPTW efficiently. They also demonstrate the contribution of agent rationality and selfishness. We believe FGAS can be adapted to solve other forms of bin-packing problems successfully. Future work includes using FGAS to solve online optimization problems, since the system is capable of creating and destroying agents as the input changes over time.

6. REFERENCES

[1] Yokoo M., Durfee E. H., Ishida T., and Kuwabara K., 1998 *Distributed constraint satisfaction problem: Formalization and algorithms*. IEEE Transactions on Data and Knowledge Engineering, 673-685

[2] Modi P. J., Shen W., Tambe M., and Yokoo M., 2003 *An asynchronous complete method for distributed constraint optimization*. Proc. Autonomous Agents and Multiagent Systems, 161-168

[3] Armstrong, A. and E. Durfee, 1997 *Dynamic Prioritization of Complex Agents in Distributed Constraint Satisfaction Problems*. Proc. 15th International Joint Conference on Artificial Intelligence, 620-625

[4] Jung H. and Tambe M., 2003 Performance Models for Large Scale Multiagent Systems: Using Distributed POMDP Building Blocks, Proc. Autonomous Agents and Multiagent Systems, 297-304

[5] Tsui K. C. and Liu J., 2003 *Multiagent Diffusion and Distributed Optimization*. Proc. Autonomous Agents and Multiagent Systems, 169-176

[6] Shehory O., Kraus S, and Yadgar O., 1999 *Emergent cooperative goal satisfaction in large-scale automated-agent systems*. Artificial Intelligence, 110(1): 1-55

[7] Lau H. C., Ono H., and Liu Q. Z., 2002 *Integrating Local Search and Network Flow to Solve the Inventory Routing Problem*. Proc. 18th National Conf. on Artificial Intelligence (AAAI), 9-14

[8] De Backer B., and Furnon V., 1997 *Meta-heuristics in Constraint Programming Experiments with Tabu Search on the Vehicle Routing Problem*, Proc. 2nd Metaheuristics International Conference

[9] Lau H. C., Lim A., and Liu Q. Z., 2000 *Solving a Supply Chain Optimization Problem Collaboratively*. Proc. 17th National Conf. on Artificial Intelligence (AAAI), 780-785

[10] Lau H. C., Lim M. K., Wan W. C., Wang H. and Wu X., 2003 *Solving Multi-Objective Multi-Constrained Optimization Problems using Hybrid Ants System and Tabu Search*. Proc. 5th Metaheuristics International Conference

[11] Lerman K. and Shehory O., 2000 *Coalition formation for largescale electronic markets.* Proc. International Conference on Multi-Agent Systems (ICMAS)

Table 1. Comparison of solution quality

Test Case	D_H	V_H	D_F	V_F	$D_{Fw/oI}$	$V_{Fw/oI}$
C201	52104	29	53654	29	69237	33
C202	53404	29	55756	29	77647	44
C203	53620	29	56901	29	79508	45
C204	54778	29	57401	28	81150	48
C205	51907	30	52827	30	56793	31
C206	50507	28	53685	28	80637	39
C207	51453	29	53935	29	79953	39
C208	52501	30	54052	30	71114	35
R201	62034	21	63538	19	72333	41
R202	56071	27	60593	20	71159	41
R203	53000	30	56550	23	72387	53
R204	49708	27	53139	19	67882	48
R205	53877	28	57088	20	70232	40
R206	52747	28	57958	20	71229	44
R207	51867	29	55271	22	71334	46
R208	49541	22	51764	19	69700	45
R209	52453	29	57455	21	68052	43
R210	52478	25	56630	21	66501	41
R211	50521	26	54781	21	68236	42
RC201	67765	28	72902	20	84361	52
RC202	67534	26	71149	21	83744	49
RC203	61722	31	66275	22	82398	47
RC204	60542	27	62255	21	86501	57
RC205	68507	27	70920	20	81217	47
RC206	64750	27	70242	22	82991	44
RC207	61677	28	66333	22	82392	49
RC208	58330	30	61994	23	78066	48
Avg			59446		75064	

Table 2. Run time Comparison

Test Case	T_H (sec)	$T_{H=F}$ (sec)	T_F (sec)	$T_{H=F}/T_F$
C201	2203.20	766.93	173.16	4.43
C202	1741.33	677.69	154.72	4.40
C203	2423.05	554.14	134.75	4.14
C204	1399.94	602.68	209.20	2.88
C205	2763.52	638.76	224.32	2.85
C206	2497.03	1077.41	277.94	3.89
C207	2539.15	685.26	287.33	2.39
C208	2278.86	790.36	199.67	3.97
R201	828.35	754.32	22.45	33.60
R202	2089.18	734.89	34.66	21.20
R203	2027.01	649.47	30.72	21.14
R204	2046.14	716.17	74.77	9.58
R205	1789.15	924.79	36.49	25.34
R206	1862.95	689.81	49.36	13.98
R207	2454.79	922.07	55.57	16.59
R208	1468.98	1000.01	114.28	8.77
R209	1918.24	602.47	39.42	15.28
R210	1812.79	828.31	48.05	17.24
R211	2821.55	652.79	72.43	9.01
RC201	2052.38	1108.78	24.65	44.98
RC202	1817.48	951.42	33.93	28.04
RC203	2581.78	1057.68	47.27	22.50
RC204	2656.73	1210.69	87.83	13.78
RC205	1221.23	852.59	26.42	32.27
RC206	2782.00	729.54	30.58	23.86
RC207	2058.50	1008.37	34.28	29.42
RC208	2848.84	758.07	66.82	11.35
Avg	2110.52	812.79	95.81	**8.48**