Information Engineering Express
International Institute of Applied Informatics
2015, Vol. 1, No. 1, 29 – 36

An agent based heuristics for large synchronized task allocation

Satoshi Takahashi [*] , Tokuro Matsuo [†]

Abstract

This paper studies a heuristics method for huge synchronized task allocation problem. In many huge task processing, it is easy to consider that their huge tasks are divided into many subtasks processed by many distributed computers. It is known that the distributed computing is fast approach when their machines process independently. However, it is necessary to communicate between each machine to solve the problem in appreciation level. We treat a snowplow problem as an example of the huge synchronized task allocation problem. We employ an agent simulation for solving the snowblower problem.

Keywords: Task allocation problem, agent heuristics, supermodular function minimization

1 Introduction

This paper studies a heuristics method for huge synchronized task allocation problem. In many huge task processing, it is easy to consider that their huge tasks are divided into many subtasks processed by many distributed computers. It is known that the distributed computing is fast approach when their machines process independently. However, it is necessary to communicate between each machine to solve the problem in appreciation level. For example, in signal processing on traffic control, each signal should communicate each other about traffic condition. many real applications cannot behave independently. It is easer to consider the independent model in the distributed computing, however, we have to remain that almost problems in the real world need to communicate and synchronize each other. We treat one of case studies for the real world's applications.This paper models a road snowplowing problem as a directed graph model, and discusses the problem's complexity. Also we propose an agent simulation heuristics algorithm for solving the problem. The snowplow problem is consisted by a task allocation problem and a scheduling problem. The task allocation problem and the scheduling problem is one of fundamental combinatorial optimization problems[1, 6]. Task allocation problems are described that m tasks are allocated to n machines such that minimizing total processing cost. This problem is applied to practical business such as staff scheduling problems and resource allocation problems. It is difficult to solve exactly when an instance is huge or objective function

[*] University of Electro-Communications, Tokyo, Japan
[†] Advanced Institute for Industrial Technology, Tokyo, Japan

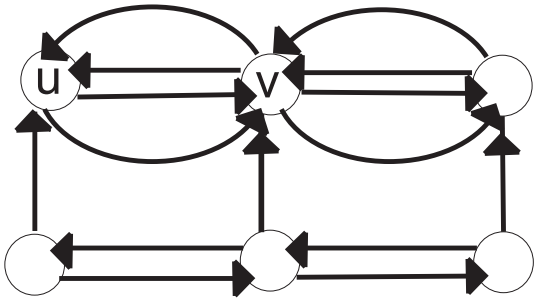

Figure 1: Directed graph model of traffic lanes

is complication. Therefore there exists several approximation and heuristics algorithm for their problem[10, 11]. The snowplow problem treated in this paper is one of task allocation problem which is needed fast computing. Snowplow tasks are very important tasks at snow covered regions. If to not be able to process huge snowplow tasks, it brings down paralysis of city functions and economic activities. Thus it is necessary to compute snowplow tasks allocation problem.

The rest of this paper consists of the following three parts. In Section 2, we describe a snowplow problem. In Section 3, we show our problem definition. In Section 4, we discuss a complexity of our problem based on mathematical programming model. And we propose a heuristics approach for our problem. Also we show the result of simple agent simulation. Finally, we present our concluding remarks and future work.

2 Snowplow Problem

Snow covered regions such as north area of japan come up against a lot of fallen snow. Because of this fallen snow, it is difficult to provide traffic services, since many roads are covered with snow. Many public administrations of snow covered regions provide a road snowplow service as public service. Snowplow service weights on their public finances, because of this fact, public administrators decay qualities of other public services. On the other hand, a quality loss of snowplow service occurs a paralysis of not only a traffic service and a public service, of but also a business activity and a development of cities. To provide efficient and high-quality snowplow service is able to hold down snowplow service fee and to apply surplus to other public services.

The snowplowing problem has many future work including modeling, since there is only few related work. E. M. Arkin et al. [2] models the snowplowing problem as optimization problem. They try to split given region in some cells by using Voronoi diagram, and propose an efficient snowplowing method on the cells. Their proposed algorithm has constant approximation ratio, but, it is not applied to our target problem.

3 Problem Definition

We treat a road snowplow tasks by using snowplow machines. First of all, we describe some definitions to consider our problem.

Definition 1. *(Snowplow region) Let $G = (V,A)$ be a directed graph, which each vertex represents cross roads. We represent a snowplow region as this directed graph. Each arc*

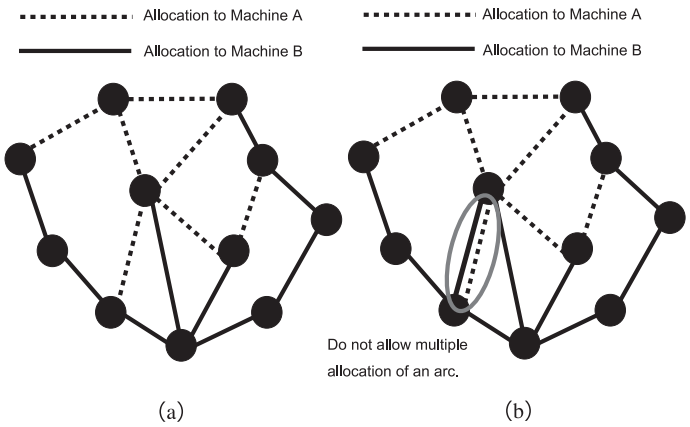

Figure 2: A task allocation based on arc-disjoint graph partition

$a \in A$ represents a traffic lane and direction of arc a represents a traveling direction. For example, when there are four traffic lanes between vertex v and u, each arc are $a_1 = (v,u)$, $a_2 = (v,u)$, $a_3 = (u,v)$ and $a_4 = (u,v)$, as figure 1.

Let $A' \subseteq A$ be a set of arcs which should be snowplowed, since we have not snowplow every arc of the graph, generally. Let $M = \{1,....,m\}$ be a set of snowplowing machines. We define a cost function of each machine i.

Definition 2 (cost function)**.** *For each machine i, a cost function* $c_i : 2^{A'} \to \mathbb{R}_+ = \{x \in \mathbb{R} \mid x \geq 0\}$ *satisfies following conditions:*

(i) $c_i(\phi) = 0$,

(ii) $\forall S \subseteq T \subseteq A'$, $c_i(S) \leq c_i(T)$,

(iii) $\forall S,T \subseteq A'$, $c_i(S) + c_i(T) \leq c_i(S \cup T) + c_i(S \cap T)$.

The condition (iii) is called as supermodularity[6, 7, 5]. There exists some traveling time between arcs a and $a' \in A'$. Hence the more increasing arcs which should be snowplowed, the more increasing its cost.

The road snowplowing problem consists from task allocation problem and scheduling problem. First of all, we consider the task allocation problem. It is inefficient way to allocate an arc to several machines, since already snowplowed arcs should not be snowplowed in a given period. Therefore we define an allocation.

Definition 3. *An allocation of snowplowing tasks to machines is a arc disjoint partitions of A', that is $S_1, S_2, ..., S_m$. Each set S_i is allocated to machine i.*

Arc-disjoint is that any arcs are only allocated one time as figure 2(a). Also arc-disjoint partition does not allow multiple allocation of an arc such as figure 2(b).

Next we consider scheduling problem of each machine. Given some feasible task allocation, we have to compute each machine's snowplowing schedule, since each machine can only move on some arcs which are not necessary to snowplow or already snowplowed arc without snowplowing. If subgraph $G[S_i]$ induced by allocated set of arcs S_i is a directed Euler graph, a machine i can estimate his snowplowing job as one stroke writing, if not, the machine i should consider other machines' behaviors. Hence, we have to compute an allocation and scheduling at the same time.

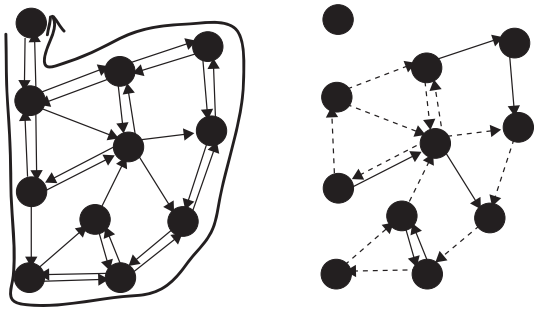

Figure 3: Graph partition based on longest path. When the algorithm found a longest path, it deletes the longest path, and update a directed graph and searches a new longest path on the updated directed graph.

4 Complexity of the road snowplowing problem

In this section, we discuss complexity of our problem. First of all, we consider a complexity of an allocation problem. We can formularize our allocation problem as

$$\text{(AP)} \quad \left| \begin{array}{ll} \text{minimize} & \sum_{i \in M} c_i(S_i) \\ \text{subject to} & \cup_{i \in M} S_i = A' \\ & S_i \cap S_j = \phi \quad \forall i \neq j. \end{array} \right.$$

The problem (AP) is a minimization problem of supermodular function. It is known that the problem as one of $\mathscr{NP}$-hard problems, because a supermodular function minimization problem is equivalent to a submodular maximization problem[3, 4]. There are some approximation algorithm proposed by [6]. The problem (AP) is not considered routing schedule. However it is hard to compute an allocation and scheduling at the same time, since (AP) is $\mathscr{NP}$-hard. Hence we try to approach that computing effective routing when an allocation is given. In this paper, we propose a heuristics algorithm to solve the problem (AP) and an agent simulation heuristics to solve routing problem.

5 Heuristics algorithm

5.1 Graph partition heuristics

An allocation of snowplowing jobs is a set of arc disjoint partitions of graph G. If a subgraph induced by every partition is an Euler graph on arc set A', then each machine can estimate his job without considering other machines' behavior. An Euler graph on arc set A' is a closed walk that traverses each arc in the partition exactly once using some arcs in $A \setminus A'$. Then each machine is allowed multiple traverse on the arc in $A \setminus A'$. Given an planer graph $G = (V, E)$, it is known an $O(|V|^3 + |E|)$ algorithm for determination of existence of given graph has arc disjoint Euler tours [9]. Hence, if there exists m arc disjoint Euler tours in the graph, each machine can work without other machines' behavior. However, Euler tour is not exist more than two in the general graph. Therefore we create another method of graph partition.

In this paper, we propose a partition method which computes a longest directed path on A' for minimization of inefficient moving. We employ a similar technique of an algorithm for a shortest path problem (figure 3). In the shortest path problem, we have a polynomial

time algorithm, Dijkstra algorithm[8]. On the other hand, we can identify a longest path problem as a shortest path problem which has negative cost on arc, however, we do not use Dijkstra algorithm to a longest path problem directly, since a graph has some cycles. If there are some cycles on the graph, Dijkstra algorithm cannot get optimal solution, since the algorithm keeps traversing the cycle forever. Hence we modified Dijkstra algorithm by adding cycle exclusion constraint. In this case, the modified algorithm is not same Dijkstra algorithm, but, it is efficient method. We can make efficient method for computing an allocation of snowplowing jobs. Next we have to make scheduling method.

5.2 Agent simulation heuristics

Our proposed graph partition heuristics do not returns Euler tours. Hence we should consider each machine's behavior, when we make a scheduling. In order to consider the machine's behavior, we employ an agent simulation. In agent simulation, each agent decides an efficient work schedule by using the given allocation. When an agent finds some inconvenient arcs, the agent exchange the arcs for another agent's allocated arcs. Each agent behaves under following rules.

Rule 1 Each agent compute minimum estimation time of allocated snowplowing jobs.

Rule 2 Each agent snowplows under directions of arcs from each start.

Rule 3 Each agent has global clock as common clock, and sends own location information to other agents in given time.

Rule 4 Each agent decides a next job by considering other agents' works. Hence the agents decide whether snowplowing or stopping at the next time.

Rule 5 When all jobs were estimated, all agents stop own works.

Rule 6 When an agent finds some inconvenient arcs, he exchange the arcs for another agent's allocated arcs.

Now we describe the agent simulation flow for snowplowing problem.

Step 1. Initialize. We give an initial placement for each agent, since in real situation all snowplow trucks do not go into action from the same place but go into action from asunder places. We allocate various edges for each agent. An agent choices his/her action based on the edge's condition.

Step 2. Counting number of snowplow arcs. Every agent checks an edges snowfall condition. If an arc $a = (i, j)$'s amount of snowfall exceeds a measure of threshold of amount of snowfall, the frag f_a is changed to true.

Step 3. Agents' behaviors. Each agent choices his/her behavior based on a condition of edges.

1. If an arc a's amount of snowfall is more than a measure of threshold, an agent snowplows on a.
2. If the agent completed the snowplow on the arc a then the flag f_a is changed to false.

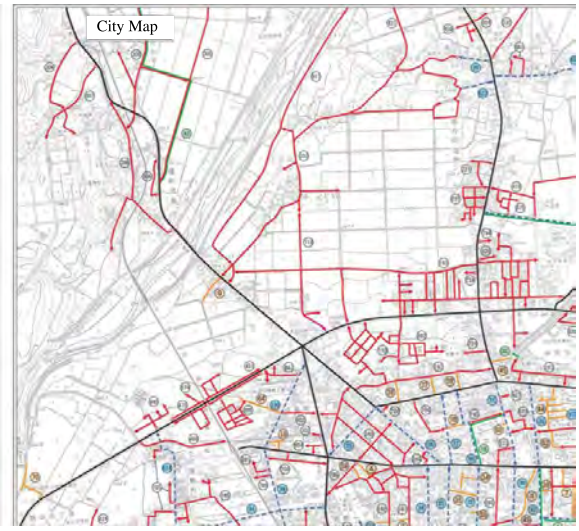

Figure 4: Region for simulation

3. Before the agent will move to next arc, he/she gather the information about other agents' places. An agent decides a next arc based on a binary variable Z and other agents' places.
4. If the agent decides the next arc as a', and its state is $Z_{a'} = 0$ then the agent snowplows this arc a'.
5. If there exists multiple arcs which are necessary to snowplow, the agent choices an arc randomly.
6. If all adjacent arcs' states are $Z_{\hat{a}} = 1$ then the agent choice a random arc and move to the edge.
7. All agents comply with a speed condition on the arc.

Step 4. Time update. We update the unit time.

Step 5. Checking terminate condition The simulation program judges whether a present elapsed time reaches the time limit. If the present time is not reached the time limit, the algorithm brings next step. Otherwise, this simulation is terminated as failure.

Step 6. Termination judge. If there exists an edge which is necessary to snowplow, the simulation program iterates from Step 2 to Step 6. Otherwise the simulation program returns a result.

5.3 Simulation result

We focus on Yonezawa city's roadmap, which is the north area city of Japan, of Figure 4 in our simulation. In this area, city employee call out about 200 snowplow trucks. Our simulation employes a snowplowing rule of the city. We show the rule as follows.

1. If an amount of snowfall is more than 10cm, then an agent must snowplow.
2. Each amount of snowfall of edges is 1cm per 18 seconds.

Table 1: The simulation result

# agents	Migration time	Total time	# agents	Migration time	Total time
1	impossibility	impossibility	14	3253	45542
2	impossibility	impossibility	15	2985	44775
3	impossibility	impossibility	16	2982	47712
4	impossibility	impossibility	17	2882	48994
5	impossibility	impossibility	18	2062	37116
6	10214	61284	19	2150	40850
7	8804	61284	20	1804	36080
8	7264	58912	21	1997	41937
9	6477	58293	22	1853	40766
10	5239	52390	23	1780	40940
11	4519	50501	24	1641	39384
12	4083	48996	25	1537	38425
13	3318	46452			

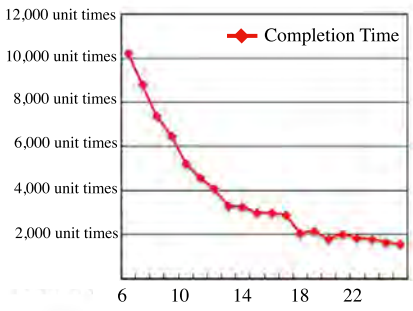

Figure 5: Shortest time of an agent's snow removal of each number of agents

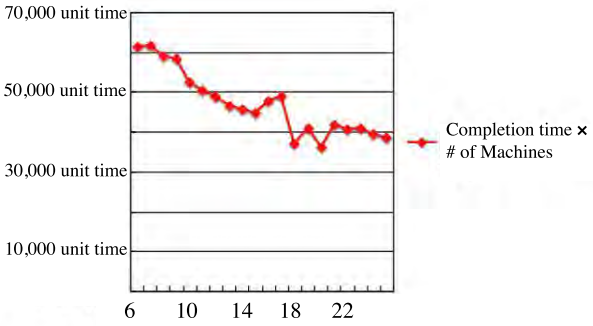

Figure 6: Total time of snow removal of each number of agents

3. Time limit is 14,400 units time (4 hours).

4. Every agent's operating speed is 36km/h on normal time.

We show a result in which the simulation changes a number of snow-plow trucks. The simulation is examined 100 times. Table 1 shows the result. "An agent's migration time" is a shortest terminated time of one agent. "Total time" is computed by multiplying "An agent's migration time" by number of agents.

Figure 5 and Figure 6 show a simulation result. A horizontal axis of Figure 5 shows a number of agents, a vertical axis shows a time in which an agent spends. This graph's curve is like $y = 1/x$. In Figure 6, a horizontal axis shows a number of agents, and a vertical axis shows a total time.

From this simulation result, it is necessary at least 6 snowplow machines to complete the snowplowing on the focused area in the time limitation. When the number of machines are grater than 13, the reduction rate is weak. From Figure 6, it is easy to show the number

of snowplow machines does not effort to the reduction of completion time.

6 Concluding and remark

This paper modeled a road snowplowing problem as a directed graph, discussed the problem's complexity. We suggested that the snowplowing problem is consisted by a task allocation problem and a scheduling problem. The task allocation problem we treated in this paper is one of $\mathscr{NP}$-hard problems. Hence, we proposed a heuristics algorithm for solving the allocation problem. Also we proposed an agent simulation heuristics algorithm for solving the scheduling problem.

References

[1] N. Katoh, and T. Ibaraki, Resource Allocation Problems. Handbook of Combinatorial Optimization (Vol. 2), D. Z. Du, and P. M. Pardalos (Eds), pp. 159–260, 1998.

[2] E. M. Arkin and M. A. Bender and J. S. B. Mitchell The snowblower problem. Proc. of 7th International Workshop on the Algorithmic Foundations of Robotics, 2006.

[3] S. Iwata, A Fully Combinatorial Algorithm for Submodular Function Minimization. Journal of Combinatorial Theory, Series B, Vol. 84, pp. 203–212, 2002.

[4] S. Iwata, and J. B. Orlin, A Simple Combinatorial Algorithm for Submodular Function Minimization. Proc. of the twentieth annual ACM-SIAM symposium on Discrete Mathematics, pp. 1230–1237, 2009.

[5] S. Fujishige, Submodular Functions and Optimization, Second Edition. Annals of Discrete Mathematics, Vol. 58, Elsevier, 2005.

[6] A. Schrijver, Combinatorial Optimization: Polyhedra and Efficiency (Algorithms and Combinatorics) . Springer varlag , 2003.

[7] B. Lehmann, and D. Lehmann, and N. Nisan, Combinatorial auctions with decreasing marginal utilities. Games and Economic Behavior, Vol. 55, pp. 270-296, 2005.

[8] R. K. Afuja, and T. L. Magnanti, and J. B. Orlin, Network Flows : Theory, Algorithms, and Applications. Prentice Hall; United States ed, 1993.

[9] S. Masuyama and S. Nakayama, What Structural Features Make Graph Problems to Have Efficient Parallel Algorithms? Using Outerplanar Graphs, Trapezoid Graphs and In-Tournament Graphs as Examples. EICE Transactions on Information and Systems, Vol. E83-D, No. 3, pp. 541–549, 2000.

[10] A. Billionnet, and M. C. Costa, and A. Sutter. An efficient algorithm for a task allocation problem. Journal of ACM, Vol. 39, pp. 502–518, 1992.

[11] Billionnet, A. and Costa, M. C. and Sutter, A. The task allocation problem with constant communication. Discrete Applied Mathematics, Vol. 131, pp. 169–177, 2003.