Title	搬送計画問題に対するネットワーク理論を利用したアプローチ
Author(s)	千葉，英史
Citation	
Issue Date	2003-03
Type	Thesis or Dissertation
Text version	author
URL	http://hdl.handle.net/10119/1655
Rights	
Description	Supervisor:浅野 哲夫，情報科学研究科，修士

An Approach Based on Network Theory to Transportation Scheduling Problem

Eishi Chiba (110078)

School of Information Science,
Japan Advanced Institute of Science and Technology

February 14, 2003

Keywords: NP-complete problem, compaction algorithm, shortest path algorithm.

Transportation Scheduling Problem is the problem of finding an optimal transportation scheduling in multi-commodity manufacture lines, which belongs to the NP-complete class. In this paper, we present a deterministic polynomial algorithm for the problem under some constraints.

A constrained transportation scheduling problem can be formulated as a linear programming problem, which can be solved using a mathematical programming software. Many problems formulated as linear programming problems can be solved by an efficient algorithm in practice. But, most of such software packages have little flexibility. So, we propose an algorithm using network theory in this paper. It has good description capability and is faster than the linear programming.

The transportation scheduling problem has been a local problem for those associated with manufacturing for a long time. But, it is gradually spotlighted. There are many differences between the past and present manufactures. The former has the feature of sequential manufacturing process. The process suited for a large amount but ony one kind of products has been used for a long time. But, there are disadvantages on the process from points of diversification of consumer needs, many customization, and generational change. The latter has the feature of the manufacturing process in multi-commodity manufacture lines. It is difficult to find an optimal transportation scheduling in the process. So, it has little taken up for an academic research. In practice, most of control programs are implemented based on experience. It is not easy to prove the optimality of the programs.

Now, a manufacturing machine must produce a certain number of manufactures in a constant time. So, we must have better transportation scheduling to get more productivity. But, most of source codes using "IF-THEN rule" and large state transition diagram are very long, and not intelligent. So it is difficult to increase productivity of softwares.

The aim of this study is to find an answer for the above problem. First, we reveal the heart of the transportation scheduling problem, formulate it as an optimization and a decision problem mathematically. Next, we prove that it is NP-complete. The proof is based on the polynomial-time reduction from the Bin-Packing Problem to the Transportation Scheduling Problem. Even if objects are ordered in their starting times, we can also prove that it is NP-complete. The proof is based on the polynomial-time reduction from the Bin-Packing Problem to the Transportation Scheduling Problem, too. Thus, we cannot expect (unless P=NP) to find an algorithm whose running time is bounded by a polynomial in the input sizes. Last, we present a deterministic polynomial algorithm when order of all events is given. The constrained problem is the one-dimensional compaction problem of input size n. Because it can be formulated as a linear programming problem, it can be solved using a mathematical programming software. Many problems formulated as linear programming problems can be solved by an efficient algorithm in practice. But, most of such software packages have little flexibility. So, we propose an algorithm based on network theory. The method has a feature that various network algorithms are used. The constrained problem can be solved in O($n$log$n$) time by using the algorithm. We implement the algorithm using LEDA. LEDA is a C++ class library, and it was developed at the Max-Plank-Institut in Saarbrucken, Germany. Because the difference between algorithm and program is small in using LEDA, we can implement the algorithm within 30 lines.