Available online at www.sciencedirect.com

ScienceDirect

Discrete Optimization 5 (2008) 97–107

DISCRETE OPTIMIZATION

www.elsevier.com/locate/disopt

Approximation algorithm for minimizing total latency in machine scheduling with deliveries

Asaf Levin[a,*], Michal Penn[b]

[a] *Department of Statistics, The Hebrew University, 91905 Jerusalem, Israel*
[b] *Faculty of Industrial Engineering and Management, Technion, Haifa, Israel*

Received 10 August 2004; received in revised form 18 November 2007; accepted 25 November 2007
Available online 31 December 2007

Abstract

We study the problem of MINIMIZING TOTAL LATENCY IN MACHINE SCHEDULING WITH DELIVERIES, which is defined as follows. There is a set of n jobs to be processed by a single machine at a plant, where job J_i is associated with its processing time and a customer i located at location i to which the job is to be delivered. In addition, there is a single uncapacitated delivery vehicle available. All jobs (vehicle) are available for processing (delivery) at time 0. Our aim is to determine the sequence in which the jobs should be processed in the plant, the departure times of the vehicle from the plant, and the routing of the vehicle, so as to minimize the total latency (job delivery time). We present a 6e $\sim$ 16.309691-approximation algorithm for the problem.
© 2007 Elsevier B.V. All rights reserved.

Keywords: Minimum latency; Scheduling; Approximation algorithm

1. Introduction

In this paper we study the problem of MINIMIZING TOTAL LATENCY IN MACHINE SCHEDULING WITH DELIVERIES (MSDL), which is defined as follows. Let $\mathcal{J} = \{J_1, J_2, \ldots, J_n\}$ be a set of n jobs to be processed by a single machine. Let $\mathcal{T} = \{t_1, t_2, \ldots, t_n\}$ be a set of n processing times and $\{1, 2, \ldots, n\}$ is a set of n locations, where customer i is located at location i for $1 \leq i \leq n$. Without loss of generality, we assume that the plant (where the machine is located) is located at 1 and $t_1 = 0$. Each job, say job J_i, is associated with its processing time t_i ($t_i \geq 0$) and its customer which is located at i. There is a single uncapacitated delivery vehicle available. After processed, job J_i needs to be delivered to its associated customer by the vehicle. We denote by c_{ij} the time it takes the vehicle to travel from location i to location j. We assume that c is a metric. All jobs are available for processing at time 0, and the vehicle is available at the plant at time 0. Our aim is to determine the sequence in which the jobs should be processed in the plant, the departure times of the vehicle from the plant, and the routing of the vehicle, so as to minimize the sum of the arrival times at the customers, that is, to minimize the total latency.

Recently there has been a growing interest and vast research on supply chain management, e.g. [15], which is of much practical importance. A supply chain contains a sequence of activities that are involved in producing

* Corresponding author.
E-mail addresses: levinas@mscc.huji.ac.il (A. Levin), mpenn@ie.technion.ac.il (M. Penn).

1572-5286/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.disopt.2007.11.007

and delivering a product or a service. Often, one will distinguish between supply chains that are manufacturing or service oriented. Clearly, the production and delivery problem discussed in this paper is an important part of many manufacturing supply chain problems.

The MSDL problem generalizes the scheduling problem of a single machine where the aim is to minimize the sum of completion times by setting $c_{ij} = 0, \forall i, j$, see Smith [14] for an algorithm and Pinedo [12] for general literature on scheduling. It also generalizes the MINIMUM LATENCY PROBLEM which is the special case of MSDL where $t_j = 0, \forall j$. The minimum latency problem is known to be NP-hard (e.g. [13]) and thus implies the NP-hardness of the MSDL. Therefore, we look for an approximation algorithm for the problem. An α-approximation algorithm produces a feasible solution with total latency no more than α times the total latency of an optimum solution to MSDL, and the value of α is called the *approximation ratio* of the algorithm.

A more general version of the MSDL problem was studied in [10]. In their setting there are n jobs and $h \leq n$ customers, customers may be associated with several jobs. Also, they allow for a capacitated vehicle while we consider the uncapacitated case. The authors present an exact dynamic programming algorithm for a fixed number of customers of complexity $O(n^{\frac{h(h+7)}{2}})$.

Our 6e $\sim$ 16.309691-approximation algorithm for the MSDL uses a function of the cost of an optimal rooted k-VERTEX AND EDGE TRAVELING SALESMAN PROBLEM, abbreviated k-VETSP, as a lower bound for the latency of the kth customer of an optimal solution. The k-VETSP is defined as follows: let $G = (V, E)$ be an undirected graph with a length function on its edges, a penalty function on its vertices, a given integer k and a prespecified vertex r. Then, the goal is to choose a subset S_r of V with $r \in S_r$ and a tour on S_r such that S_r contains at least k vertices and the sum of the costs of the vertices and edges of the tour is minimized. Our MSDL-approximation algorithm follows similar arguments to the ones used by Goemans and Kleinberg [7] for the minimum latency problem. Now, if we had 2-approximate k-VETSP's for all $k = 2, \ldots, n$, then we could have run our MSDL-approximation algorithm to obtain a 6e-approximated solution. We use a similar trick to the one used by Archer et al. [1] for the minimum latency problem, to successfully bluff our MSDL-approximation algorithm. We pretend to have 2-approximate k-VETSP's for all k's by interpolating the costs of the tours for the missing values of k. We refer to these as phantom tours. We then prove by using an approximation version of Megiddo's parametric search method [11] (see also [9]), that if our MSDL-approximation algorithm was to run with both real and phantom tours, it will never choose any of the phantom tours, so it will never discover our bluff.

To derive our MSDL-approximation algorithm we utilize a $(2 - \frac{1}{n-1})$-k-VETSP-approximation algorithm, for some k's, $1 \leq k \leq n$, that are not under our control. The latter algorithm follows similar lines as used by Garg [5] and Chudak's et al. [4] in their k-MST-approximation algorithms, where in the k-MST problem one seeks for a minimum tree spanning at least k vertices. Our $(2 - \frac{1}{n-1})$-k-VETSP-approximation algorithm is a Lagrangean relaxation algorithm and employs, as a subroutine, Goemans and Williamson's primal-dual-approximation algorithm [8] for the rooted PRIZE-COLLECTING TRAVELING SALESMAN problem (PCTSP). In the PCTSP problem, the aim is to find S, a subset of V not including r, and a tour spanning the rest of the vertices not in S, so as to minimize the cost of the edges in the tour plus the cost of the vertices in S (penalties for vertices that are not spanned by the tour).

The k-VETSP generalizes the k-TSP problem which is the special case of the k-VETSP where all vertex costs are zero. We note that some of the approximation results for the k-TSP, e.g. [2,5], are used to derive our simple $(2 + \epsilon)$-approximation algorithm for the k-VETSP for all k's.

The rest of the paper is organized as follows. In Section 2 we give some definitions and present our $(2 - \frac{1}{n-1})$-k-VETSP-approximation algorithm for some k's that are not under our control. Since the proof of the correctness of the algorithm needs to elaborate on the PCTSP problem, it is postponed to Section 5. In Section 3 we construct our 16.309691-MSDL-approximation algorithm assuming the existence of a 2-approximation algorithm for k-VETSP for all k's. In Section 4 we remove the above assumption and present the main result of this paper, namely, a 6e $\sim$ 16.309691-approximation algorithm for the MSDL. As indicated above, Section 5 is devoted to the correctness proof of our $(2 - \frac{1}{n-1})$-k-VETSP-approximation algorithm for some k's. We conclude by drawing few directions for further research in Section 6.

2. Definitions and preliminaries

We use graph notations where $G = (V, E)$ stands for the complete undirected graph with n vertices, over the vertex set $V = \{1, 2, \ldots, n\}$ of the locations. Also, there is a non-negative length c_e on each edge $e \in E$, and a non-negative