1 Introduction

We consider the problem of scheduling trucks at a transshipment terminal. Each truck either delivers or picks up a certain amount of goods. Of course, a truck that is supposed to pick up goods cannot proceed if the current inventory level is less than the amount the truck should pick up. We restrict ourselves to the case where there is only a single gate at the transshipment terminal where trucks can be processed. Several objectives can be of interest in the context of a transshipment terminal. First, we consider the weighted flow time, that is the sum of weighted truck completion times is to be minimized. Second, we consider a due date for each truck and aim at minimizing either the maximum lateness or the number of tardy trucks.

Inventory constraints as described above have been considered in project scheduling, see Neumann and Schwindt [4] for example. It has been shown that inventory constraints are a generalization of both renewable and non-renewable resources if the number of goods is not limited. To the best of our knowledge only a few papers consider scheduling of transport vehicles to be served at a transshipment terminal. Scheduling of trucks at cross-dock-terminals has been considered in Boysen et al. [1]. A stylized model is introduced where two gates of the terminal are considered. Boysen et al. [1] show that minimizing makespan is strongly NP-hard even if all processing times are equal. Yu and Egbelu [5] consider a similar model and develop a heuristic to solve the problem.
2 Problem Specification

The truck scheduling problem can be specified by input and output in terms of machine scheduling problem. We represent trucks by a set $J$ of $n = |J|$ jobs. We distinguish between two classes of jobs. Jobs in $J^+$ represent delivering trucks while jobs in $J^-$ represent pick-up trucks. Note that $J = J^+ \cup J^-$ and $J^+ \cap J^- = \emptyset$. Furthermore, we denote $n^+ = |J^+|$ and $n^- = |J^-|$. We have a processing time $p_j$ and, possibly, a due date $d_j$ and a weight $w_j$ given for each job $j \in J$. Moreover, $\delta_j$ specifies the amount by which the inventory is changed when job $j \in J$ is processed. Note that $\delta_j > 0$ if $j \in J^+$ and $\delta_j < 0$ if $j \in J^-$. Additionally, an initial inventory $I_0$ is given.

A solution is a sequence $\sigma$ such that each job $j \in J$ is contained in $\sigma$ exactly once. Let $\sigma(s)$ and $I_s$ denote the $s$th job in $\sigma$ and the inventory level immediately after $\sigma(s)$ has been processed, that is $I_s = I_{s-1} + \delta_{\sigma(s)}$. A solution $\sigma$ is a feasible solution if $I_s \geq 0$ for each $s \in \{1, \ldots, n\}$.

We derive completion time $C_j$ of job $j = \sigma(s)$ according to a solution $\sigma$ as $C_j = \sum_{t=1}^{s} p_{\sigma(t)}$. If due date $d_j$ is given for each job $j \in J$, then we consider $L_j = C_j - d_j$ to be the lateness of job $j$. If $L_j > 0$, then we say that $j$ is tardy. Now, problems we consider in the paper at hand can be defined as follows.

- Find a feasible solution minimizing the total weighted completion time $\sum w_jC_j$. We denote this problem extending the notation introduced in Graham et al. [2] by $1|\text{inv}|\sum w_jC_j$.

- Find a feasible solution minimizing the maximum lateness $L_{\text{max}} = \max_{j \in J} L_j$. We denote this problem by $1|\text{inv}|L_{\text{max}}$.

- Find a feasible solution minimizing the number of tardy jobs. We denote this problem by $1|\text{inv}|\sum U_j$.

In order to have a convenient notation for special cases we introduce binary parameters $C^p(J'), C^w(J'), C^d(J'), \text{ and } C^\delta(J')$ indicating whether $p_j = p$, $w_j = w$, $d_j = d$, and $\delta_j = \delta$, respectively, for each $j \in J'$ (parameter’s value equals 1) or not (parameter’s value equals 0). For example, $C^p(J^+) = 1$ means that all jobs in $J^+$ have equal processing time.

3 Results

**Theorem 1** $1|\text{inv}|\sum w_jC_j$ is strongly NP-hard even for $w_j = w$.

**Theorem 2** $1|\text{inv}|L_{\text{max}}$ is strongly NP-hard.

The proofs can be done by reduction from 3-PARTITION.

**Theorem 3** If $C^p(J^+) + C^w(J^+) + C^\delta(J^+) \geq 2$ and $C^p(J^-) + C^w(J^-) + C^\delta(J^-) \geq 2$, then $1|\text{inv}|\sum w_jC_j$ is solvable in $O(n \log n)$ time.

We develop an algorithm that sorts $J^-$ and $J^+$ first and merge them to the optimal sequence afterwards.

**Theorem 4** If $C^p(J^+) + C^\delta(J^+) \geq 1$, then $1|\text{inv}|L_{\text{max}}$ can be solved in $O(n^3)$ time.
We develop an algorithm using binary search on $L_{\text{max}}$ and solving the corresponding feasibility subproblem.

**Theorem 5** If $C^d(J^+) = C^p(J^-) = 1$ and $C^p(J^+) + C^\delta(J^+) \geq 1$, then $1|\text{inv}| \sum U_j$ can be solved in $O(n^2 \log n)$ time.

We develop an algorithm which solves iteratively subproblems where the number of early jobs of $J^+$ is bounded from below.

**Theorem 6** If $C^d(J^+) = 1$ and $C^p(J^+) + C^\delta(J^+) \geq 1$, then $1|\text{inv}| \sum U_j$ can be solved in $O \left( n^2 \left( \sum |\delta_j| \right)^2 \left( \sum p_j \right)^3 \right)$ time.

Again we iteratively consider subproblems where the number of early jobs of $J^+$ is bounded from below. We reduce this subproblem to the shortest path problem.

**Theorem 7** If $C^d(J^-) = C^p(J^+) = 1$ and $C^p(J^-) + C^\delta(J^-) \geq 1$, then $1|\text{inv}| \sum U_j$ can be solved in $O(n^6)$ time.

We iteratively consider subproblems where the number of early jobs of $J^-$ is bounded from below. We reduce the subproblem to the restricted shortest path problem which can be solved in polynomial time in our case, see Hassin [3].

Summarizing, we determine complexity of all special cases of $1|\text{inv}| \sum C_j$ and $1|\text{inv}| L_{\text{max}}$ as well as most special cases of $1|\text{inv}| \sum w_j C_j$ and $1|\text{inv}| \sum U_j$. However, some cases remain open, for example $1|\text{inv}| \sum w_j C_j$ with $C^p(J^+) = C^w(J^+) = C^\delta(J^+) = C^\delta(J^-) = 1$ or $C^p(J^-) = C^w(J^-) = C^\delta(J^-) = C^\delta(J^+) = 1$ is minimal open.

**References**


