Combinatorial Structure of Single machine rescheduling problem

Yuan Jinjiang

Department of mathematics, Zhengzhou University
Zhengzhou, Henan 450052

yuanjj@zzu.edu.cn

1 Introduction and Problem Formulation

• Rescheduling, just as its name implies, means to schedule the jobs again, together with a set of new jobs.

• In the rescheduling on a single machine, a set of original jobs has already been scheduled to minimize some cost objective, when a new set of jobs arrives and creates a disruption. The decision maker needs to insert the new jobs into the existing schedule without excessively disrupting it.

Home Page
Title Page
◀◀ ▶▶
◀ ▶
Page 2 of 18
Go Back
Full Screen
Close
Quit

• By Hall and Potts (2004), the rescheduling problem for jobs on a single machine can be stated as follows.

Let $\mathcal{J}_O = \{J_1, ..., J_{n_O}\}$ denote a set of original jobs to be processed non-preemptively on a single machine.

In the model, we assume that the jobs in $\mathcal{J}_O$ have been scheduled optimally to minimize some classical objective and that π^* is an optimal schedule.

Let $\mathcal{J}_N = \{J_{n_O+1}, ..., J_{n_O+n_N}\}$ denote a set of new jobs.

Write $\mathcal{J} = \mathcal{J}_O \cup \mathcal{J}_N$.

Each job $J_j \in \mathcal{J}$ has an integral processing time $p_j \geq 0$, an integer release date $r_j \geq 0$ and an integer due date d_j.

We assume that the new jobs' information (processing times and release dates) becomes known at time zero after a schedule for the jobs of J_O has been determined, but before processing begins.

Let $n = n_O + n_N$.

Let π^* and σ^* denote an optimal schedule of the jobs of $\mathcal{J}_O$ and $\mathcal{J}$, respectively.

- •For any schedule σ of the jobs in $\mathcal{J}$, we define the following variables:
- $S_j(\sigma)$ is the time at which job $J_j \in \mathcal{J}$ starts its processing.
- $C_j(\sigma) = S_j(\sigma) + p_j$ is the time at which job $J_j \in \mathcal{J}$ is completed.
- $C_{\max}(\sigma) = \max\{C_j(\sigma)\}$ is the makespan of jobs in $\mathcal{J}$ under the schedule σ.
- $D_j(\pi^*, \sigma)$ is the sequence disruption of job $J_j \in \mathcal{J}_O$, i.e., if J_j is the x-th job in π^* and the y-th job in σ, respectively, then $D_j(\pi^*, \sigma) = |y - x|$.
- $\Delta_j(\pi^*, \sigma) = |C_j(\sigma) - C_j(\pi^*)|$ is the time disruption of job $J_j \in \mathcal{J}_O$.

Here the sequence disruption of job $J_j \in \mathcal{J}_O$ in schedule σ is the absolute value of the difference between the positions of that job in σ and π^*.

When there is no ambiguity, the above five parameters are simplified to C_j, $C_{\max}$, $D_j(\pi^*)$, and $\Delta_j(\pi^*)$, respectively.

- The standard classification scheme for scheduling problems is a three-field classification $\alpha|\beta|\gamma$, where α indicates the scheduling environment, β describes the job characteristics or restrictive requirements, and γ defines the optimality criterion.

  Here we consider only single-machine problems, thus implying that $\alpha = 1$.

  Under β, we indicate a constraint on the amount of disruption where applicable.

  Such constraints include the following four forms:

- $D_{\max}(\pi^*) \leq k$: $\max_{J_j \in \mathcal{J}_O}\{D_j(\pi^*)\} \leq k$, the maximum sequence disruption of the jobs cannot exceed k.

- $\sum D_j(\pi^*) \leq k$: $\sum_{J_j \in \mathcal{J}_O} D_j(\pi^*) \leq k$, the total sequence disruption of the jobs cannot exceed k.

- $\Delta_{\max}(\pi^*) \leq k$: $\max_{J_j \in \mathcal{J}_O}\{\Delta_j(\pi^*)\} \leq k$, the maximum time disruption of the jobs cannot exceed k.

- $\sum \Delta_j(\pi^*) \leq k$: $\sum_{J_j \in \mathcal{J}_O} \Delta_j(\pi^*) \leq k$, the total time disruption of the jobs cannot exceed k.

Introduction and . . .

The existing . . .

$1|r_j, D_{\max}(\pi^*) \leq \ldots$

Home Page

Title Page

◀◀ ▶▶

◀ ▶

Page 5 of 18

Go Back

Full Screen

Close

Quit

Let f be the cost function to be minimized.

The scheduling problems are of the following forms:

- $1|D_{\max}(\pi^*) \leq k|f$
- $1|\sum D_j(\pi^*) \leq k|f$
- $1|\Delta_{\max}(\pi^*) \leq k|f$
- $1|\sum \Delta_j(\pi^*) \leq k|f$.

For a job set $\mathcal{E} \subseteq \mathcal{J}$, a schedule σ of the jobs in $\mathcal{E} \subseteq \mathcal{J}$ is called regular for $\mathcal{E}$ if there is no other schedule h such that $C_j(h) \leq C_j(\sigma)$ for every job J_j and there is at lease one job J_i such that $C_i(h) < C_i(\sigma)$.

The jobs in a regular schedule are said to be regularly scheduled.

●● We will only consider optimal schedules, since there must be an optimal schedule which is regular when $f = f(C_1, ..., C_n)$ is non-decreasing for each C_j.

Home Page
Title Page

Go Back
Full Screen
Close
Quit

2 The existing complexity results and open problems

- Polynomially solved problems:

$1\|\Delta_{\max}(\pi^*) \leq k\|L_{\max}$,	$O(n + n_N \log n_N)$,	Hall and Potts (OR, 2004)
$1\|D_{\max}(\pi^*) \leq k\|L_{\max}$,	$O(n + n_N \log n_N)$,	Hall and Potts (OR, 2004)
$1\|\Delta_{\max}(\pi^*) \leq k\|\sum C_j$,	$O(n + n_N \log n_N)$,	Hall and Potts (OR, 2004)
$1\|D_{\max}(\pi^*) \leq k\|\sum C_j$,	$O(n + n_N \log n_N)$,	Hall and Potts (OR, 2004)
$1\|\sum D_j(\pi^*) \leq k\|\sum C_j$,	$O(n_O^2 n_N^2)$,	Hall and Potts (OR, 2004)
$1\|r_j, D_{\max}(\pi^*) \leq k\|C_{\max}$,	$O(n_N^2(n_O + n_N))$,	Yuan and Mu (Accepted by EJOR, 2006)
$1\|r_j, \sum D_j(\pi^*) \leq k\|C_{\max}$.	polynomial time,	Yuan and Mu (In submission, 2006)

- **NP-hard problems:**

$1\|\sum \Delta_j(\pi^*) \le k\|L_{\max}$,	strongly NP-hard,	Hall and Potts (OR, 2004)
$1\|\sum \Delta_j(\pi^*) \le k\|\sum C_j$,	$O(n_O n_N \min\{n_O P_N, n_N P_O\})$,	Hall and Potts (OR, 2004)
$1\|r_j, \Delta_{\max}(\pi^*) \le k\|C_{\max}$,	strongly NP-hard,	Yuan and Mu (2006)
$1\|r_j, \sum \Delta_j(\pi^*) \le k\|C_{\max}$,	strongly NP-hard,	Yuan and Mu (2006)

- **Open problems:**

$1|\sum D_j(\pi^*) \le k|L_{\max}$. See Hall and Potts (OR, 2004).

3 Combinatorial Structure of Optimal Solutions

The polynomial-time algorithms of the above problems are based on the well described combinatorial structure of optimal solutions.

The following are some well-known job sequence.

SPT sequence: In which, the jobs are sequenced according to the shortesr processing time first rule:

$$p_1 \leq p_2 \leq ... \leq p_n.$$

EDD sequence: In which, the jobs are sequenced according to the earliest due date first rule:

$$d_1 \leq d_2 \leq ... \leq d_n.$$

ERD sequence: In which, the jobs are sequenced according to the earliest release date first rule:

$$r_1 \leq r_2 \leq ... \leq r_n.$$

Theorem 1 (Hall and Potts, 2004) For problem $1|\Delta_{\max}(\pi^*) \le k|L_{\max}$, there is an optimal schedule of the (EDD, EDD) property, i.e., the jobs in $\mathcal{J}_O$ are sequenced in ERD order, and the jobs in $\mathcal{J}_N$ are sequenced in ERD order too.

From the above property, an $O(n + n_N \log n_N)$ time algorithm are designed to solve the problem $1|\Delta_{\max}(\pi^*) \le k|L_{\max}$.

Theorem 2 (Hall and Potts, 2004) For problem $1|D_{\max}(\pi^*) \leq k|L_{\max}$, there is an optimal schedule of the (EDD, EDD) property, i.e., the jobs in $\mathcal{J}_O$ are sequenced in ERD order, and the jobs in $\mathcal{J}_N$ are sequenced in ERD order too.

From the above property, an $O(n + n_N \log n_N)$ time algorithm are designed to solve the problem $1|D_{\max}(\pi^*) \leq k|L_{\max}$.

Theorem 3 (Hall and Potts, 2004) For problem $1|\Delta_{\max}(\pi^*) \leq k| \sum C_j$, there is an optimal schedule of the (SPT, SPT) property, i.e., the jobs in $\mathcal{J}_O$ are sequenced in SPT order, and the jobs in $\mathcal{J}_N$ are sequenced in SPT order too.

From the above property, an $O(n + n_N \log n_N)$ time algorithm are designed to solve the problem $1|\Delta_{\max}(\pi^*) \leq k| \sum C_j$.

Theorem 4 (Hall and Potts, 2004) For problem $1|D_{\max}(\pi^*) \leq k|\sum C_j$, there is an optimal schedule of the (SPT, SPT) property, i.e., the jobs in $\mathcal{J}_O$ are sequenced in SPT order, and the jobs in $\mathcal{J}_N$ are sequenced in SPT order too.

From the above property, an $O(n + n_N \log n_N)$ time algorithm are designed to solve the problem $1|D_{\max}(\pi^*) \leq k|\sum C_j$.

Home Page

Title Page

◀◀ ▶▶

◀ ▶

Go Back

Full Screen

Close

Quit

Theorem 5 (Hall and Potts, 2004) For problem $1|\sum D_j(\pi^*) \leq k|\sum C_j$, there is an optimal schedule of the (SPT, SPT) property, i.e., the jobs in $\mathcal{J}_O$ are sequenced in SPT order, and the jobs in $\mathcal{J}_N$ are sequenced in SPT order too.

From the above property, an $O(n_O^2 n_N^2)$ time algorithm are designed to solve the problem $1|\sum D_j(\pi^*) \leq k|\sum C_j$.

Theorem 6 (Yuan and Mu, 2005) For problem $1|r_j, D_{\max}(\pi^*) \leq k|C_{\max}$, there is an optimal schedule of the weak ERD property, i.e., the jobs in $\mathcal{J}_O$ are sequenced in ERD order; and furthermore, the inserted jobs in $\mathcal{J}_N$ is a maximum weighted basis of a matroid.

From the above property, an $O(n_N^2(n_O + n_N))$ time algorithm are designed to solve the problem $1|r_j, D_{\max}(\pi^*) \leq k|C_{\max}$.

Introduction and . . .
The existing . . .
$1|r_j, D_{\max}(\pi^*) \leq \ldots$
Home Page
Title Page
◀◀ ▶▶
◀ ▶
Page 15 of 18
Go Back
Full Screen
Close
Quit

For the problem $1|r_j, \sum D_j(\pi^*) \leq k|C_{\max}$, we consider its dual problem

$$1|r_j, C_{\max} \leq Y| \sum D_j(\pi^*).$$

The above two problems have the same decision version

$$1|r_j| \sum D_j(\pi^*) \leq k, C_{\max} \leq Y,$$

which whether there is a schedule π such that

$$\sum D_j(\pi^*, \pi) \leq k, \text{ and } C_{\max}(\pi) \leq Y.$$

Theorem 7 If the dual problem $1|r_j, C_{\max} \leq Y| \sum D_j(\pi^*)$ can be solved in $O(F(n))$ time, then the original problem can be solved in $O(F(n) \log M)$ time by using $\log M$ time binary searches, where M is length of the period of $C_{\max}$, for example, we can choose $M = r_{\max}$.

Theorem 8 (Yuan and Mu, 2006) For dual problem $1|r_j, C_{\max} \leq Y| \sum D_j(\pi^*)$, there is an optimal schedule of the the following three properties:

(1) The weak ERD property, i.e., the jobs in $\mathcal{J}_O$ are sequenced in ERD order;

(2) The starting time of the first job is $S = Y - \sum p_j$ and there are no idle time in $[S, Y)$;

(3) At each time moment when the machine can be used and there are no original job is released, a released new jobs of maximum processing time is processed.

From the above property, an $O(n \log n + n_N^2)$ time algorithm are designed to solve the dual problem $1|r_j, C_{\max} \leq Y| \sum D_j(\pi^*)$.

Consequently, the problem $1|r_j, \sum D_j(\pi^*) \leq k|C_{\max}$ can be solved in $O((n \log n + n_N^2) \log r_{\max})$ time.

Thank You!