Available online at www.sciencedirect.com
ScienceDirect
Theoretical Computer Science 362 (2006) 238–247
Theoretical Computer Science
www.elsevier.com/locate/tcs

Utilization of nonclairvoyant online schedules

Mohamed Eid Hussein[a], Uwe Schwiegelshohn[b,*]

[a] *Mathematics Department of South Valley University, 81528, Aswan, Egypt*
[b] *Robotics Research Institute, University Dortmund, 44221 Dortmund, Germany*

Received 20 August 2004; received in revised form 4 October 2005; accepted 27 June 2006

Communicated by A. Fiat

Abstract

This paper addresses the analysis of nondelay, nonpreemptive, nonclairvoyant online schedules for independent jobs on m identical machines. In our online model, all jobs are submitted over time. We show that the commonly used makespan criterion is not well suited to describe utilization for this online problem. Therefore, we directly address utilization and determine the maximum deviation from the optimal utilization for the given scheduling problem.

Keywords: Online scheduling; Utilization

1. Introduction

Consider the following problem: George owns m servers and uses them to deliver web services $j \in \tau$. Due to performance and security constraints, a server executes only a single service at a time. George gets paid for the time a web service is occupying a server even if the web service is interactive and the server is partially idle due to a slow user response. As the individual behaviour is highly dynamic George's scheduling system does not allow any reservations but uses a simple first-come-first-serve approach. George observes high average utilization of his servers during core business hours while there are only few customers at night, that is, most customers are not willing to postpone their activities to off-business hours but rather switch to another provider if they see that George's servers are occupied. Now, George likes to know whether it pays off to install additional servers.

In this example, customers can monitor the availability of servers. They may submit jobs even if all servers are busy. In this case, the job is queued. But of course, there may be other customers who decide not to submit a job when no idle server is available. Therefore, George cannot estimate the additional server capacity that is requested by customers. He can only identify two simple cases: on the one hand, if all servers are always or almost always busy during core business hours and if there are additional customer requests, George's revenue can most likely be increased by installing another server. On the other hand, if at least one server is always idle, George can save maintenance expenses by retiring a server. However, typically there will be periods where all servers are busy alternating with periods of idle servers during core business hours. Is such a situation due to missing overall customer demand or due to the inability of the fair

* Corresponding author. Tel.: +49 23 17552634; fax: +49 23 17553251.
E-mail addresses: memhussein2003@yahoo.com (M.E. Hussein), uwe.schwiegelshohn@udo.edu (U. Schwiegelshohn).

0304-3975/$ - see front matter © 2006 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2006.06.027

scheduling method to handle demand peaks caused by an unfortunate arrival sequence of jobs? How much idleness in the schedule should be tolerated?

Unfortunately, George does not know much about those jobs executed on his servers. Neither the submission time r_j of a job j is announced in advance nor the servers have any knowledge of the remaining processing time of a running job as this may be at the discretion of the customer. Without any of this information about the jobs, no intelligent scheduling strategy can be employed. Forcing the customers to provide the processing times p_j of their jobs in advance may be a hassle to most customers and is typically of little help as experiments with users of parallel computers have shown that those estimates are very unreliable, even for batch jobs [4]. The use of statistic workload models can only provide some benefits if the dynamics of the process are not too high. In addition, it is expensive to generate those models and to tailor scheduling algorithms accordingly.

Therefore, George wants to determine first the influence of his simple online scheduling method. To this end, he likes to compare the achieved utilization of his servers with the utilization that would be produced by an optimal scheduling algorithm having a perfect oracle. Once he knows the maximum deviation from the optimal utilization, George may consider using a more complex approach based on a statistical workload model.

Formally, we have a job system τ consisting of independent jobs that must be scheduled on m identical machines without preemption. Each job $j \in \tau$ is characterized by its processing time $p_j > 0$ and its release date $r_j \geqslant 0$. Further, a job is not known before it is released (*release over time*) and its processing time is only determined once the job has completed (*nonclairvoyant scheduling*). Contrary to some other online scheduling problems [1], no assignment of a waiting job to a machine must be made before the machine is actually available. We denote the completion time of job j in a schedule S by $C_j(S)$. Schedule S is legal if no machine is executing more than a single job at any time, no job starts before its release date r_j, and each job is fully processed without interruption. Therefore, job j starts at time $C_j(S) - p_j$ in schedule S. Due to those conditions, it only makes sense to employ *nondelay* schedules [7], that is, no machine is kept idle while a job is waiting for processing.

We have not yet mentioned the evaluation criterion. Together with a brief review of previous related work, this will be the main subject of the next section. To provide appropriate tools for the evaluation, we then define a basic job system and a basic nondelay schedule. Then we show that those basic job systems suffice to determine the worst-case ratio for utilization. Finally, in Section 4, we derive the maximum deviation of any nondelay schedule from the optimal utilization for a given job system.

2. Makespan and utilization

First, we formally define the utilization within interval $[t_1, t_2)$ of schedule S by

$$U_{[t_1,t_2)}(S, \tau) = \sum_{j\in\tau} \max\{0, \min\{t_2, C_j(S)\} - \max\{t_1, C_j(S) - p_j\}\},$$

that is the amount of machine resources used by job system τ within this interval. This formal definition agrees with the intuitive understanding most people have about utilization. Clearly, jobs that are partially processed within the interval $[t_1, t_2)$ only partially contribute to the utilization of the interval. Note that this definition can also be applied in a case where τ is only a subset of all jobs in S. In this case, utilization is restricted to the jobs from this subset which are processed within the given interval. Without restriction of generality, we assume that the start time of the interval is 0, that is, for a job system τ and its schedule S, we want to maximize the utilization $U_{[0,t^*)}(S, \tau)$ for a fixed target time t^*. The optimal utilization of job system τ within the interval is denoted by $U^*_{[0,t^*)}(\tau)$. Note that there may be jobs j with $r_j \leqslant t^*$ that are processed before time t^* in a schedule with optimal utilization while they start after time t^* in schedule S. Those jobs contribute to $U^*_{[0,t^*)}(\tau)$ but not to $U_{[0,t^*)}(S, \tau)$.

For parallel machines, the makespan criterion $C_{\max}(S) = \max_{j\in\tau} C_j(S)$ is usually considered to represent utilization of a schedule S, see, for instance, Pinedo's book [7]. Commonly, it is taken for granted that a schedule with a smaller makespan has a higher utilization. But this relation does not always hold in the case of a target interval. The simple example of Fig. 1 shows that in the interval [0, 5), a schedule with the optimal makespan may have a worse utilization than a schedule with a nonoptimal makespan.

Further, it is easy to show that $C_{\max}(S)/C^*_{\max} \leqslant 2 - 1/m$ holds with S being an arbitrary nondelay schedule and $C^*_{\max}$ denoting the optimal makespan, see Shmoys et al. [9]. This factor is tight. According to this result, George must assume

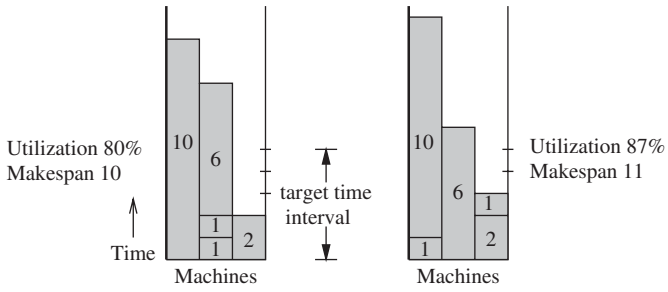

Fig. 1. Comparison of Makespan and Utilization for 2 Schedules.

Table 1
Comparison between utilization and makespan for two job systems

	$\frac{U^*_{[0,m)}(\tau_i)}{U_{[0,m)}(S_i,\tau_i)}$	$\frac{C_{\max}(S_i)}{C^*_{\max}(\tau_i)}$
$i=1$	$\frac{m^2}{m^2-m+2} < 1+\frac{1}{m}$	$2-\frac{2}{m}$
$i=2$	$1+\frac{1}{3}$	1.5

that the selection process may be responsible for up to 50% idleness. It is one objective of this paper to determine whether George's assumption is correct.

Finally, we quantitatively compare the worst case ratio (competitive factor) of the makespan criterion with the utilization criterion for two job systems τ_1 and τ_2 executed on an even number of machines m. All jobs are released at time 0 and the target time is m. In job system τ_1, we have m short jobs of processing time 1 and m long jobs of processing time $m-1$. In the nondelay schedule S_1, we start $m-1$ long jobs at time 0 and execute all short jobs on a single machine in the interval $[0, m)$. This requires a single long job to start at time $m-1$ and to run until time $2m-2$. In the optimal schedule, all long jobs start at time 0 and all short jobs are executed in parallel in the interval $[m-1, m)$.

In job system τ_2, there are $m^2/2$ jobs of processing time 1 and $m/2$ jobs of processing time m. In the nondelay schedule S_2, we execute all short jobs in the interval $[0, m/2)$ using all machines and start all long jobs at time $m/2$ while in the optimal schedule, all long jobs start at time 0 and all short jobs are processed in the interval $[0, m)$ using only $m/2$ machines. Note that in all schedules, all jobs are started in the target time interval $[0, m)$. The results in Table 1 demonstrate that there is no quantitative relation between both criteria.

Therefore, the makespan criterion may not always be an appropriate criterion to represent utilization in all scheduling problems.

Nonclairvoyant scheduling has been investigated first by Motwani et al. [5]. It is mainly used in connection with preemptive scheduling, resource augmentation, see Kalyanasundaram and Pruhs [2], and flow time minimization, see Kalyanasundaram and Pruhs [3]. Competitive factors have been determined for many online scheduling problems, see Sgall's survey [8]. However, we are not aware of any work that is closely related to the problem addressed in this paper. It may only be worth mentioning Naroska and Schwiegelshohn [6] who showed that the competitive factor $2-1/m$ also holds for the nonclairvoyant online makespan problem if the jobs are parallel, that is, if a job requires more than one machine to be concurrently available. But to our knowledge, no theoretical analysis has been provided for the utilization criterion yet.

3. Basic job systems

We start with some definitions and notations that are used in the later sections. We call an interval $[t_a, t_b)$ of a schedule *S fully utilized* or simply *full* if all m machines are busy executing jobs at any time instant of this interval. A full interval is *max-full* if it is not a true subset of another full interval in this schedule. Note that in any max-full interval $[t_a, t_b)$ of

a nondelay schedule, at least one job is released and starts at time t_a. For a given schedule S, we define $[t_a(S), t_b(S))$ to be the last max-full interval of S. If S has no full interval then we set $t_a(S) = t_b(S) = 0$.

Now, we introduce a basic job system. We will later show that the worst-case ratio is produced by such a job system.

Definition 1. A job system τ is called a *basic* job system if there is a nondelay schedule S for τ such that the following conditions are valid for any max-full interval $[t_a, t_b)$ of S and a fixed $\varepsilon > 0$:
1. $C_j(S) - p_j > r_j + (t_b - t_a)$ holds for all jobs $j \in \tau$ with $r_j < t_a < r_j + p_j$.
2. $p_j \leqslant \varepsilon$ holds for all $C_j(S) - p_j \in [t_a, t_b)$.
3. $r_j = t_a$ holds for all jobs $C_j(S) - p_j \in [t_a, t_b]$.

Schedule S is called a *basic* nondelay schedule.

Every job of basic job system τ with $p_i \leqslant \varepsilon$ is called a *short* job while all other jobs are *long* jobs. In a basic nondelay schedule, long jobs are either started at their release dates or immediately after a max-full interval while short jobs are also started within a max-full interval. Without restriction of generality we can assume that all jobs start in order of their release dates in a basic nondelay schedule, that is, job j_1 does not start after job j_2 if $r_{j_1} < r_{j_2}$ holds.

Consider a nondelay schedule in which all long jobs of a basic job system start at their release date. We say that such a schedule is a *utilization* schedule of this basic job system.

Property 1 guarantees that such a utilization schedule exists for each basic job system, as shown in the next corollary.

Corollary 2. *In a utilization schedule, at most $m - 1$ long jobs of a basic job system τ can execute concurrently.*

Proof. Clearly, at most $m - 1$ long jobs of τ execute concurrently in the corresponding basic nondelay schedule. Therefore, we only need to consider the situation when long jobs do not start at their release dates in a basic nondelay schedule, that is, when they start at the end of a max-full interval.

Let $[t_a, t_b)$ be a max-full interval of the basic nondelay schedule of τ. Due to Property 1 of Definition 1, for every long job $j \in \tau$ with $r_j < t_a < r_j + p_j$, we have $C_j(S) > r_j + p_j + t_b - t_a > t_a + t_b - t_a = t_b$. Therefore, those jobs cannot complete at or before time t_b in the basic nondelay schedule. Therefore, at most $m - 1$ long jobs can execute concurrently in the utilization schedule at time t_a. □

Consider two machines and a time interval $[t_1, t_2)$ of the basic nondelay schedule, such that only short jobs of the same release date are executed at those machines within this interval. Due to Property 2 of Definition 1, the makespans of both machines within this interval differ at most by ε.

Property 3 generates the maximum amount of flexibility for the optimal schedule thus increasing the worst case ratio between the utilization of a schedule and the optimal utilization.

Next, we want to show that any job system τ' with a nondelay schedule S' can be transformed into a basic job system τ with a basic nondelay schedule S such that we have $U^*_{[0,t)}(\tau')/U_{[0,t)}(S', \tau') \leqslant U^*_{[0,t)}(\tau)/U_{[0,t)}(S, \tau)$ for all $t \leqslant t^*$. t^* denoting an arbitrary but fixed target time. Then it is sufficient to only consider basic job systems and basic nondelay schedules for the purpose of a worst case analysis. To this end, we first discuss a simple modification of a given job system.

Corollary 3. *Let τ' be a job system and S' be a schedule for τ' on m identical machines. Job system τ is generated from τ' by dividing an arbitrary job $j \in \tau'$ into two jobs j_1 and j_2 such that $0 < p_{j_1} < p_j$, $r_{j_1} = r_j$, $p_{j_2} = p_j - p_{j_1}$, and $r_{j_2} = r_j + p_{j_1}$ hold. Schedule S is derived from schedule S' by simply starting job j_1 instead of job j and starting job j_2 immediately after the completion of job j_1.*

Then the inequality $U^*_{[0,t)}(\tau')/U_{[0,t)}(S', \tau') \leqslant U^*_{[0,t)}(\tau)/U_{[0,t)}(S, \tau)$ holds for all $t \leqslant t^*$.

Proof. Note that the completion time of each job $j' \neq \{j, j_1, j_2\}$ is identical in both schedules S and S'. Further, the release date r_{j_2} of job j_2 does not interfere with scheduling j_2 immediately after the completion of j_1 if schedule S' is legal.

Therefore, $C_{j_1}(S) = C_j(S') - p_{j_2}$ and $C_{j_2}(S) = C_j(S')$ hold and S is a legal schedule as no job starts before its release date and no machine is used to execute two jobs at the same time.

Clearly, we have $U_{[0,t)}(S,\tau) = U_{[0,t)}(S',\tau')$ for all $t \leqslant t^*$. $U^*_{[0,t)}(\tau) \geqslant U^*_{[0,t)}(\tau')$ holds for all $t \leqslant t^*$ as the splitting of job j cannot decrease $U^*_{[0,t)}(\tau')$ for any $t \leqslant t^*$. This leads to

$$\frac{U^*_{[0,t)}(\tau')}{U_{[0,t)}(S',\tau')} \leqslant \frac{U^*_{[0,t)}(\tau)}{U_{[0,t)}(S,\tau)} \quad \text{for all } t \leqslant t^*. \qquad \square$$

Splitting a job within or at the end of a full interval produces again a nondelay schedule if the original schedule was already a nondelay schedule.

Now, we transform an arbitrary job system and its nondelay schedule into a basic job system and its basic nondelay schedule.

Corollary 4. *For any job system τ', a nondelay schedule S', and an arbitrary but fixed $\varepsilon > 0$, there is a basic job system τ and a basic nondelay schedule S such that $U^*_{[0,t)}(\tau')/U_{[0,t)}(S',\tau') \leqslant U^*_{[0,t)}(\tau)/U_{[0,t)}(S,\tau)$ holds for all $t \leqslant t^*$.*

Proof. Consider a max-full interval $[t_a, t_b)$ in schedule S'.

Property 1. We assume a job $j \in \tau'$ with $r_j < t_a < r_j + p_j$ and $C_j(S') - p_j \leqslant r_j + (t_b - t_a)$, that is, Property 1 of Definition 1 is violated by j. Therefore, we have $C_j(S') - p_j - r_j + t_a \leqslant t_b$. The inequality $r_j \leqslant C_j(S') - p_j$ leads to $t_a \leqslant C_j(S') - r_j - p_j + t_a$, that is, $C_j(S') - r_j - p_j + t_a \in [t_a, t_b]$. Similarly, we obtain immediately $C_j(S') > C_j(S') - r_j - p_j + t_a > C_j(S') - p_j$, that is, job j is processed in schedule S' at time instance $C_j(S') - r_j - p_j + t_a$. Hence, we split job j in schedule S' at time $C_j(S') - r_j - p_j + t_a$ by using Corollary 3.

Property 2. We repeatedly apply Corollary 3 to all jobs j starting in $[t_a, t_b)$ with $p_j > \varepsilon$ such that $p_{j_1} \leqslant min\, \varepsilon, t_b - C_j + p_j$ holds.

Property 3. The smallest release date of all jobs starting in interval $[t_a, t_b]$ is t_a as schedule S' is a nondelay schedule. Therefore, we set the release date of all jobs starting in $[t_a, t_b]$ to t_a. Clearly, the resulting schedule will again be nondelay.

The same transformations are applied to all other max-full intervals of S'. As each transformation cannot decrease the ratio $U^*_{[0,t)}(\tau')/U_{[0,t)}(S',\tau')$ for any $t \leqslant t^*$, we have $U^*_{[0,t)}(\tau')/U_{[0,t)}(S',\tau') \leqslant U^*_{[0,t)}(\tau)/U_{[0,t)}(S,\tau)$ for all $t \leqslant t^*$ for the resulting basic job system τ and its basic nondelay schedule S. $\square$

Next assume a decreasing sequence of ε_i with $\lim_{i\to\infty} \varepsilon_i = 0$. Then Corollary 4 produces a sequence of basic job systems τ_i from a given job system τ'. For those basic job systems, we consider the utilization schedules. In the following corollary, we show that the utilization of those utilization schedules approaches the optimal utilization for all $t \leqslant t^*$ if $i \to \infty$.

Corollary 5. *Let τ_i be an infinite sequence of basic job systems τ_i derived from job system τ' such that $\lim_{i\to\infty} \varepsilon_i = 0$. For all corresponding utilization schedule S_i, $\lim_{i\to\infty} U_{[0,t)}(S_i,\tau_i) = U^*_{[0,t)}(\tau_i)$ holds for all $t \leqslant t^*$.*

Proof. Only short jobs that start within or at the end of a max-full interval of the utilization schedule can produce a deviation from the optimal utilization. If schedule S_i has k max-full intervals then $U_{[0,t)}(S_i,\tau_i) - U^*_{[0,t)}(\tau_i) < m \cdot k \cdot \varepsilon$ holds for all $t \leqslant t^*$ due to Property 2 of Definition 1. Therefore, we have $\lim_{i\to\infty} U_{[0,t)}(S_i,\tau_i) = \lim_{\varepsilon\to 0} U_{[0,t)}(S_i,\tau_i) = U^*_{[0,t)}(\tau_i)$ for all $t \leqslant t^*$. $\square$

Therefore, we assume a sufficiently small ε and say that utilization schedule of a basic job system is an optimal schedule. As we need only one optimal schedule for our worst case analysis we can ignore all other optimal schedules for these job systems if they exist. Without restriction of generality we assume that in such an optimal schedule, all short jobs also start in order of their release dates. But note that in a utilization schedule, a long job with a higher release date can start before a short job with a lower release date.

4. Utilization range

This section proves the main result of our paper. Due to the results of the previous section, we only need to analyze basic job systems with a sufficiently small ε, their basic nondelay schedules, and their utilization schedules in order to determine a worst case deviation. Without mentioning it explicitly, we assume that ε is always small enough.

For such a basic job system τ, its basic nondelay schedule S and its optimal schedule σ, we want to determine the difference

$$D_t(S,\tau) = U^*_{[0,t)}(\tau) - U_{[0,t)}(S,\tau) = U_{[0,t)}(\sigma,\tau) - U_{[0,t)}(S,\tau) = U_{[t,\infty)}(S,\tau) - U_{[t,\infty)}(\sigma,\tau)$$

for each time instant $t \leqslant t^*$. Intuitively, this value describes the machine resources that are not busy executing jobs from τ in schedule S before time t while they are used to process jobs from τ before time t in the optimal schedule. From this definition, we obtain the following relation for $t' < t$:

$$D_t(S,\tau) = D_{t'}(S,\tau) + U_{[t',t)}(\sigma,\tau) - U_{[t',t)}(S,\tau). \tag{1}$$

An upper bound of $D_t(S,\tau)$ for a basic job system τ is given by the following lemma.

Lemma 6. *$D_t(S,\tau) \leqslant \frac{1}{4}U^*_{[0,t)}(\tau)$ holds for each basic job system τ with a sufficiently small ε, its basic nondelay schedule S, and every time instant $0 < t \leqslant t^*$.*

Proof. We prove this lemma by induction on the number k of different release dates. The lemma trivially holds for $k = 0$, that is, if τ is empty. Therefore, we assume that it is true for all basic job systems with at most k different release dates. Then we consider a basic job system τ with $k+1$ different release dates.

We define the last release date $r = \max\{r_j \mid j \in \tau\}$, the set of all jobs with the last release date $\tau_r = \{j \in \tau \mid r_j = r\}$, and the set of all other jobs $\tau_k = \{j \in \tau \mid r_j < r\} = \tau \backslash \tau_r$. Note that τ_k is a basic job system with k different release dates and that every job $j \in \tau_k$ starts before time r in the basic nondelay schedule S. Further, we define the time instances $t_S = \max\{r, t_b(S)\}$ and $t_\sigma = \max\{r, t_b(\sigma)\}$. Schedules σ and σ_k are the utilization schedules for basic job systems τ and τ_k, respectively, while S_k is the basic nondelay schedule for τ_k. Schedules σ and σ_k are identical in the time interval $[0, r)$ as no job $j \in \tau_r$ can start before time r in any schedule and all long jobs start at their release dates in both schedules. Similarly, schedules S and S_k are identical for all jobs in τ_k. The main part of the proof is divided into three steps:

Step 1: We show that it is sufficient to determine $D_{t_\sigma}(S,\tau)$. Due to the last statement of the previous paragraph and our induction assumption, there is

$$D_t(S,\tau) = D_t(S,\tau_k) = D_t(S_k,\tau_k) \leqslant \tfrac{1}{4}U^*_{[0,t)}(\tau_k) = \tfrac{1}{4}U^*_{[0,t)}(\tau) \quad \textit{for all } t \leqslant r.$$

As no additional jobs are released after time r and S is a nondelay schedule, there are at least as many machines idle at time t_2 as at time t_1 in schedule S with $r \leqslant t_1 < t_2$.

Let $[t_1, t_2)$ be a subinterval of $[r, t_\sigma)$ such that no machine becomes idle in (t_1, t_2) of schedule S and let the number of busy machines in $[t_1, t_2)$ be m_{t_1}. Then we have

$$\begin{aligned} U_{[0,t_2)}(\sigma,\tau) &= U_{[0,t_1)}(\sigma,\tau) + U_{[t_1,t_2)}(\sigma,\tau) \\ &= U_{[0,t_1)}(\sigma,\tau) + m(t_2 - t_1) \end{aligned}$$

and

$$\begin{aligned} D_{t_2}(S,\tau) &= D_{t_1}(S,\tau) + U_{[t_1,t_2)}(\sigma,\tau) - U_{[t_1,t_2)}(S,\tau) \\ &= D_{t_1}(S,\tau) + m(t_2 - t_1) - m_{t_1}(t_2 - t_1). \end{aligned}$$

This leads to

$$\frac{D_{t_2}(S,\tau)}{U_{[0,t_2)}(\sigma,\tau)} = \frac{D_{t_1}(S,\tau) + (m - m_{t_1})(t_2 - t_1)}{U_{[0,t_1)}(\sigma,\tau) + m(t_2 - t_1)}.$$

Therefore, exactly one of the two following sequences of inequalities is valid:

$$\frac{D_{t_1}(S,\tau)}{U_{[0,t_1)}(\sigma,\tau)} < \frac{D_{t_2}(S,\tau)}{U_{[0,t_2)}(\sigma,\tau)} < 1 - \frac{m_{t_1}}{m}$$

or

$$\frac{D_{t_1}(S,\tau)}{U_{[0,t_1)}(\sigma,\tau)} \geqslant \frac{D_{t_2}(S,\tau)}{U_{[0,t_2)}(\sigma,\tau)} \geqslant 1 - \frac{m_{t_1}}{m}$$

As m_{t_1} is decreasing monotonically with growing t_1 in the interval $[r, t_\sigma)$ this results in

$$\max_{t\in[r,t_\sigma)} \left\{ \frac{D_t(S,\tau)}{U_{[0,t)}(\sigma,\tau)} \right\} = \max \left\{ \frac{D_r(S,\tau)}{U_{[0,r)}(\sigma,\tau)}, \frac{D_{t_\sigma}(S,\tau)}{U_{[0,t_\sigma)}(\sigma,\tau)} \right\}.$$

Further, we have $U_{[t_\sigma,t)}(S,\tau) \geqslant U_{[t_\sigma,t)}(\sigma,\tau)$ with $t > t_\sigma$ as no long job can complete earlier in schedule S than in the optimal schedule σ and no job starts at time t_σ or later in schedule σ. This leads to

$$D_t(S,\tau) = D_{t_\sigma}(S,\tau) + U_{[t_\sigma,t)}(\sigma,\tau) - U_{[t_\sigma,t)}(S,\tau) \leqslant D_{t_\sigma}(S,\tau) \text{ for all } t > t_\sigma.$$

Step 2: We assume $t_b(\sigma_k) \leqslant r$.

$t_\sigma \leqslant t_S$ immediately leads to $D_{t_\sigma}(S,\tau)/U_{[0,t_\sigma)}(\sigma,\tau) \leqslant D_r(S,\tau)/U_{[0,r)}(\sigma,\tau)$. Hence, we assume that $t_\sigma > t_S > 0$ holds and have

$$U_{[t_1,t_2)}(S,\tau_k) = U_{[t_1,t_2)}(S_k,\tau_k) \geqslant U_{[t_1,t_2)}(\sigma_k,\tau_k) = U_{[t_1,t_2)}(\sigma,\tau_k) \quad \text{for all } r \leqslant t_1 < t_2$$

as no short job from τ_k completes after time r in schedules σ and σ_k due to the assumption of this step.

We define

$$m_\sigma = \frac{U_{[t_\sigma,\infty)}(S,\tau_r) - U_{[t_\sigma,\infty)}(\sigma,\tau_r)}{t_S - r}.$$

For each long job $j \in \tau_r$, its contribution to the term $U_{[t_\sigma,\infty)}(S,\tau_r) - U_{[t_\sigma,\infty)}(\sigma,\tau_r)$ is upper bounded by $t_S - r$ as all those jobs start at time t_S in schedule S and at time r in schedule σ, respectively. Therefore, the value m_σ is at most the number of long jobs from τ_r that complete after time t_σ in schedule S. Hence, at most the machine time product $(t_\sigma - t_S)(m - m_\sigma)$ can be idle in time interval $[r, t_\sigma)$ in schedule S. This results in

$$\begin{aligned} m(t_\sigma - r) &= U_{[r,t_\sigma)}(\sigma,\tau) \\ &= U_{[r,t_\sigma)}(\sigma,\tau_k) + U_{[r,t_\sigma)}(\sigma,\tau_r) \\ &= U_{[r,t_\sigma)}(\sigma,\tau_k) + \sum_{j\in\tau_r} p_j - U_{[t_\sigma,\infty)}(\sigma,\tau_r) \end{aligned}$$

and

$$m(t_\sigma - r) \leqslant U_{[r,t_\sigma)}(S,\tau_k) + \sum_{j\in\tau_r} p_j - U_{[t_\sigma,\infty)}(S,\tau_r) + (t_\sigma - t_S)(m - m_\sigma).$$

With the definition of m_σ, this leads to the inequality

$$\begin{aligned} U_{[r,t_\sigma)}(\sigma,\tau_k) + (t_S - r)m_\sigma &= U_{[r,t_\sigma)}(\sigma,\tau_k) + U_{[t_\sigma,\infty)}(S,\tau_r) - U_{[t_\sigma,\infty)}(\sigma,\tau_r) \\ &\leqslant U_{[r,t_\sigma)}(S,\tau_k) + (t_\sigma - t_S)(m - m_\sigma) \end{aligned}$$

and yields

$$m_\sigma \leqslant \frac{(t_\sigma - t_S)m + \Delta}{t_\sigma - r} \tag{2}$$

with

$$\Delta = U_{[r,t_\sigma)}(S,\tau_k) - U_{[r,t_\sigma)}(\sigma,\tau_k) = D_r(S,\tau_k) - D_{t_\sigma}(S,\tau_k),$$

see Eq. (1). Remember that all jobs from τ_k must start before r in schedules σ and S. This produces the following chain of inequalities:

$$m(t_\sigma - r) > U_{[r,t_\sigma)}(S, \tau_k) \geqslant U_{[r,t_\sigma)}(S, \tau_k) - U_{[r,t_\sigma)}(\sigma, \tau_k) = \Delta \geqslant 0.$$

From Inequality 2, we obtain

$$(t_S - r)m_\sigma \leqslant \frac{(t_S - r)(t_\sigma - t_S)m + \Delta(t_S - r)}{t_\sigma - r}.$$

The right-hand side of this inequality becomes maximal for $t_S = (t_\sigma + r)/2 + \Delta/2m$. This results in

$$(t_S - r)m_\sigma \leqslant \frac{m(t_\sigma - r)}{4} + \frac{\Delta}{2}\left(1 + \frac{\Delta}{2m(t_\sigma - r)}\right) = \frac{1}{4}U_{[r,t_\sigma)}(\sigma, \tau) + \frac{\Delta}{2}\left(1 + \frac{\Delta}{2m(t_\sigma - r)}\right).$$

Finally, we obtain

$$\begin{aligned} D_{t_\sigma}(S, \tau) &= U_{[t_\sigma,\infty)}(S, \tau_k) + U_{[t_\sigma,\infty)}(S, \tau_r) - U_{[t_\sigma,\infty)}(\sigma, \tau_k) - U_{[t_\sigma,\infty)}(\sigma, \tau_r) \\ &= D_{t_\sigma}(S, \tau_k) + U_{[t_\sigma,\infty)}(S, \tau_r) - U_{[t_\sigma,\infty)}(\sigma, \tau_r) \\ &= D_{t_\sigma}(S, \tau_k) + (t_S - r)m_\sigma \\ &= D_r(S, \tau_k) - \Delta + (t_S - r)m_\sigma \leqslant \frac{1}{4}U^*_{[0,r)}(\tau) - \Delta + \frac{1}{4}U_{[r,t_\sigma)}(\sigma, \tau) + \frac{\Delta}{2}\left(1 + \frac{\Delta}{2m(t_\sigma - r)}\right) \\ &= \frac{1}{4}U^*_{[0,t_\sigma)}(\tau) - \frac{\Delta}{2}\left(1 - \frac{\Delta}{2m(t_\sigma - r)}\right) \leqslant \frac{1}{4}U^*_{[0,t_\sigma)}(\tau). \end{aligned}$$

Step 3: Alternatively, we assume $t_b(\sigma_k) > r$.

In this step, we transform basic job system τ into another basic job system τ' with the basic nondelay schedule S' such that $D_{t_\sigma}(S, \tau) \leqslant D_{t_\sigma}(S', \tau')$ and $U^*_{[0,t_\sigma)}(\tau) \geqslant U^*_{[0,t_\sigma)}(\tau')$ hold and τ' either has only k different release dates or the situation of Step 2 applies to τ'. To this end, we distinguish four cases:

1. There is no long job in τ_r or $t_S = r$ holds, that is, all long jobs with release date r start at time r in schedule S. Then we have $D_{t_\sigma}(S, \tau) = D_{t_\sigma}(S, \tau_k)$ and $U^*_{[0,t_\sigma)}(\tau) \geqslant U^*_{[0,t_\sigma)}(\tau_k)$ as no short job completes after t_σ in both schedules S and σ, and $C_j(S) = C_j(\sigma)$ holds for all long jobs $j \in \tau_r$. This leads to

$$\frac{D_{t_\sigma}(S, \tau)}{U^*_{[0,t_\sigma)}(\tau)} \leqslant \frac{D_{t_\sigma}(S, \tau_k)}{U^*_{[0,t_\sigma)}(\tau_k)}.$$

2. $C_j(S) \leqslant t_\sigma$ holds for a long job $j \in \tau_r$. Then job j is split into short jobs with release date r. This transformation results in job system τ' and schedules S' and σ' with $U_{[t_\sigma,\infty)}(S', \tau') \geqslant U_{[t_\sigma,\infty)}(S, \tau)$ and $U_{[t_\sigma,\infty)}(\sigma', \tau') = U_{[t_\sigma,\infty)}(\sigma, \tau)$. Hence, we have

$$\frac{D_{t_\sigma}(S', \tau')}{U^*_{[0,t_\sigma)}(\tau')} \geqslant \frac{D_{t_\sigma}(S, \tau)}{U^*_{[0,t_\sigma)}(\tau)}.$$

3. $t_S > r$ holds and there are long jobs $j_1 \in \tau_r$ with $C_{j_1}(S) > t_\sigma$ and $j_2 \in \tau_k$ with $t_S \leqslant C_{j_2}(S) \leqslant t_\sigma$. Then we create τ' from τ by replacing j_1 and j_2 with jobs j'_1 and j'_2 such that $r_{j'_1} = r_{j_1} = r$, $p_{j'_1} = \max\{p_{j_2} + r_{j_2} - r, 0\}$, $r_{j'_2} = r_{j_2}$, and $p_{j'_2} = p_{j_1} + p_{j_2} - p_{j'_1} = \min\{r - r_{j_2}, p_{j_2}\} + p_{j_1}$. Note that there is no job j_1 if the transformation leads to $p_{j_1} = 0$. Further, the starting times of job j_2 in S and job j'_2 in the new basic nondelay schedule S' are identical, that is, we have $C_{j_2}(S) - p_{j_2} = C_{j'_2}(S') - p_{j'_2}$. Similarly, there is $C_{j_1}(S) - p_{j_1} = C_{j'_1}(S') - p_{j'_1} = t_S$. In the new optimal schedule σ', we obtain

$$C_{j'_1}(\sigma') = r_{j'_1} + p_{j'_1} = \max\left\{C_{j_2}(\sigma), r\right\}$$

and

$$C_{j'_2}(\sigma') = r_{j'_2} + p_{j'_2} = \min\{r, r_{j_2} + p_{j_2}\} + p_{j_1} = \min\{C_{j_1}(\sigma), C_{j_2}(\sigma) + p_{j_1}\}$$

which leads to $U_{[t_\sigma,\infty)}(\sigma', \tau') \leqslant U_{[t_\sigma,\infty)}(\sigma, \tau)$ due to $C_{j_2}(\sigma) \leqslant C_{j_2}(S) \leqslant t_\sigma$. Further remember that $t_S - r < C_{j_2}(S) - p_{j_2} - r_{j_2}$ follows directly from Property 1 of Definition 1. This leads to

$$\begin{aligned} C_{j_2'}(S') &= C_{j_2}(S) - p_{j_2} + p_{j_2'} \\ &= C_{j_2}(S) + p_{j_1} + \min\{r - r_{j_2} - p_{j_2}, 0\} \geqslant t_S + p_{j_1} = C_{j_1}(S) \end{aligned}$$

and

$$\begin{aligned} C_{j_1'}(S') &= C_{j_1}(S) - p_{j_1} + p_{j_1'} \\ &= C_{j_1}(S) + p_{j_2} - p_{j_2'} \\ &= C_{j_1}(S) + C_{j_2}(S) - C_{j_2'}(S') \leqslant C_{j_2}(S). \end{aligned}$$

Therefore, we obtain $U_{[t_\sigma,\infty)}(S', \tau') \geqslant U_{[t_\sigma,\infty)}(S, \tau)$. This results in

$$\frac{D_{t_\sigma}(S', \tau')}{U^*_{[0,t_\sigma)}(\tau')} = 1 - \frac{U_{[0,t_\sigma)}}{(S', \tau')} U^*_{[0,t_\sigma)}(\tau') \geqslant 1 - \frac{U_{[0,t_\sigma)}}{(S, \tau)} U^*_{[0,t_\sigma)}(\tau) = \frac{D_{t_\sigma}(S, \tau)}{U^*_{[0,t_\sigma)}(\tau)}.$$

4. $t_S > r$ holds and there is no long job $j \in \tau$ with $t_S \leqslant C_j(S) \leqslant t_\sigma$. Let m_S be the number of machines that are idle in the interval $[t_S, t_\sigma)$ of schedule S, and let m_r be the number of short jobs from τ_r that start at time r in schedule S.
   (a) $m_S \leqslant m_r$: We split every long job $j \in \tau_r$ into a long job j' with $p_{j'} = p_j - \varepsilon$ and an additional short job. Both jobs have release date $r' = r + \varepsilon$. Then we increase the release date of all short jobs from τ_r to r' as well. This results in job system τ', basic nondelay schedule S' with $t_{S'} = t_b(S')$ and utilization schedule σ'. Due to $m_S \leqslant m_r$, we have $t_{S'} \geqslant t_S + \varepsilon$, $U_{[t_\sigma,\infty)}(S, \tau) \leqslant U_{[t_\sigma,\infty)}(S', \tau')$, and $U_{[t_\sigma,\infty)}(\sigma, \tau) = U_{[t_\sigma,\infty)}(\sigma', \tau')$. This yields
   $$\frac{D_{t_\sigma}(S', \tau')}{U^*_{[0,t_\sigma)}(\tau')} \geqslant \frac{D_{t_\sigma}(S, \tau)}{U^*_{[0,t_\sigma)}(\tau)}.$$
   This process is repeated until $r' = t_b(\sigma_k)$ (start situation of Step 2) holds.
   (b) $m_S > m_r$: We combine every long job from τ_r with a short job from τ_r with processing time ε. The resulting jobs all have release date $r' = r - \varepsilon$. Further, we decrease the release date of all other short jobs from τ_r to r' as well. This leads to $U_{[t_\sigma,\infty)}(\sigma', \tau') = U_{[t_\sigma,\infty)}(\sigma, \tau)$ for the new utilization schedule σ'. Due to $m_S > m_r$, $t_{S'} \geqslant t_S - \varepsilon$ holds for the new basic job system τ' and its basic nondelay schedule S'. This results in $U_{[t_\sigma,\infty)}(S', \tau') \geqslant U_{[t_\sigma,\infty)}(S, \tau)$. Therefore, we have
   $$\frac{D_{t_\sigma}(S', \tau')}{U^*_{[0,t_\sigma)}(\tau')} \geqslant \frac{D_{t_\sigma}(S, \tau)}{U^*_{[0,t_\sigma)}(\tau)}.$$
   Together with the transformation of Case 3, this process is repeated until we have $t_{S'} = r'$ (Case 1) or the basic job system τ' has only k different release dates. □

With this result, we can finally determine an upper bound for $U^*_{[0,t)}(\tau)/U_{[0,t)}(S, \tau)$ if schedule S is a nondelay schedule.

Theorem 7. *For any job system τ and a nondelay schedule S for τ on m identical machines, the inequality $U^*_{[0,t)}(\tau)/U_{[0,t)}(S, \tau) \leqslant \frac{4}{3}$ holds for all $0 < t \leqslant t^*$. This bound is tight.*

Proof. Due to Corollary 4, we only need to consider basic job systems with a sufficiently small ε and their basic nondelay schedules. Let τ be such a basic job system and S be its basic nondelay schedule.

Lemma 6 yields $D_t(S, \tau) = U^*_{[0,t)}(\tau) - U_{[0,t)}(S, \tau) \leqslant \frac{1}{4} U^*_{[0,t)}(\tau)$ for all $0 < t \leqslant t^*$. This immediately results in

$$\frac{U^*_{[0,t)}(\tau)}{U_{[0,t)}(S, \tau)} \leqslant \frac{4}{3}.$$

Finally, assume $m > 1$ machines with m being even. Our job system τ contains $m/2$ jobs of processing time 2 and m jobs of processing time 1, all with release date 0. In schedule S, all m jobs with processing time 1 start concurrently at time 0 while the longer jobs all start at time 1. Clearly, S is a nondelay schedule and $U_{[0,2)}(S, \tau) = 1.5m$ holds.

In the optimal schedule σ, all $m/2$ jobs of processing time 2 and $m/2$ of the other jobs start concurrently at time 0 while the remaining jobs of processing time 1 start at time 1. This results in $U_{[0,2)}(\sigma, \tau) = U^*_{[0,2)}(\tau) = 2m$ and $U^*_{[0,2)}(\tau)/U_{[0,2)}(S, \tau) = \frac{4}{3}$. □

5. Concluding remarks

In this paper, we showed that the makespan criterion is not well suited to quantitatively describe utilization for our online problem. Therefore, we used the criterion utilization directly and gave a worst case analysis for nondelay schedules with this criterion. Due to this analysis, up to 25% idle time in the schedule may be due to the selected order of the job execution while the corresponding value provided by the makespan criterion is 50%.

Also, the analysis suggested that a longest-job-first-strategy may be appropriate in a corresponding offline problem. This confirms the small approximation factor achieved by this strategy for similar makespan problems [7]. However, no analysis for the utilization criterion is available for offline problems. This may be subject of further research.

References

[1] S. Albers, Better bounds for online scheduling, SIAM J. Comput. 29 (2) (1999) 459–473.
[2] B. Kalyanasundaram, K. Pruhs, Speed is as powerful as clairvoyance, J. ACM 47 (4) (2000) 217–243.
[3] B. Kalyanasundaram, K. Pruhs, Minimizing flow time nonclairvoyantly, J. ACM 50 (4) (2003) 551–567.
[4] C.B. Lee, Y.Schwartzman, J.Hardy, A. Snavely, Are user runtime estimates inherently inaccurate? in: Proc 10th Workshop on Job Scheduling Strategies for Parallel Processing, June 2004, pp. 153–161.
[5] R. Motwani, S. Phillips, E. Torng, Non-clairvoyant scheduling, Theoret. Comput. Sci. 130 (1994) 17–47.
[6] E. Naroska, U. Schwiegelshohn, On an online scheduling problem for parallel jobs, Inform. Process. Lett. 81 (6) (2002) 297–304.
[7] M. Pinedo, Scheduling: Theory, Algorithms and Systems, second ed., Prentice-Hall, Englewood Cliffs, NJ, 2002.
[8] J. Sgall, On-line scheduling—a survey, in: A. Fiat, G.J. Woeginger (Eds.), In Online Algorithms: The State the Art, Lecture Notes in Computer Science, Springer, Berlin, Vol. 1442, 1998, pp. 196–231.
[9] D. Shmoys, J. Wein, D. Williamson, Scheduling parallel machines on-line, SIAM J. Comput. 24 (6) (1995) 1313–1331.