Worst-Case Performance of Two Critical Path Type Algorithms

Gaurav Singh and Yakov Zinder

University of Technology, Sydney.

Abstract

The critical path method remains one of the most popular approaches in practical scheduling. This paper characterizes this method by the worst-case analysis of Hu's algorithm and the Brucker-Garey-Johnson algorithm. Both algorithms are based on the idea of the critical path method and were developed for the model with unit execution time operations, precedence constraints and parallel identical processors. We present a tight upper bound for the Brucker-Garey-Johnson algorithm and tight performance guarantees for Hu's algorithm. The previously known bound for Hu's algorithm derived by N.F. Chen (Chen 1975) is only asymptotically achievable. It is also shown that in some sense Chen's bound cannot be generalized for the Brucker-Garey-Johnson algorithm.

Keywords: Scheduling Theory, Identical Processors, Unit Execution Times, Maximum Lateness, Precedence Constraints, Worst-Case Analysis, Critical Path Method.

AMS Subject classification: 90B35.

1 Introduction.

Suppose that a set N={1,2,...,n} of n operations is to be processed on $m > 1$ identical processors subject to precedence constraints in the form of an anti-reflexive, anti-symmetric and transitive relation α on N. If operation a precedes operation b in the partially ordered set (N, α), or in other words, the ordered pair $(a, b) \in \alpha$, then operation a must be processed before operation b can begin its processing. It is convenient to represent a partially ordered set of operations by an acyclic directed graph where nodes correspond to the operations and the arcs reflect the precedence constraints. Each operation can be processed on any processor, and once a processor begins executing an operation then it continues until completion (i.e. no preemptions are allowed). The processing starts at time $t = 0$. Each operation $a \in N$ has a due date d_a, the desired point in time by which the processing should be completed.

Each processor can process at most one operation at a time. All the operations have the same processing time which, without loss of generality, can be taken as the unit of time.

Since no preemptions are allowed, to specify a schedule s it suffices to designate for each operation $a \in N$ a processor and its completion time $C_a(s)$. If operation a has a completion time $C_a(s)$, then its processing commences at time $C_a(s) - 1$. Since the processing of the operations commences at $t = 0$ and each operation requires one unit of time, without loss of generality, we will consider only schedules in which all completion times are positive integers. The goal is to find a schedule which minimizes the criterion of maximum lateness

$$L_{max}(s) = \max_{a \in N}(C_a(s) - d_a).$$

If all due dates are equal to zero this problem converts to the makespan problem with the criterion

$$C_{max}(s) = \max_{a \in N} C_a(s).$$

The makespan problem, and therefore the maximum lateness problem, is NP-hard (Ullman 1975). Several polynomial time algorithms were developed for various particular cases. The first of these results was obtained by T.C. Hu (Hu 1961) who showed that the well known critical path method yields an optimal schedule for the makespan problem if the graph corresponding to the partially ordered set (N, α) is an in-tree. For an arbitrary partially ordered set of operations Hu's algorithm gives only an approximate solution. It became standard practice to characterize approximation scheduling algorithms by performance guarantees. For example we refer the reader to (Lawler et al. 1993), and (Parker 1995). For Hu's algorithm Chen and Liu (Chen 1975), and (Chen and Liu 1975) established the following performance guarantee:

$$C_{max}(s^{CP}) \leq \begin{cases} \frac{4}{3} C_{max}(s^*) & \text{for } m = 2 \\ \left(2 - \frac{1}{m-1}\right) C_{max}(s^*) & \text{for } m \geq 3\ , \end{cases} \tag{1}$$

where s^{CP} is a schedule obtained by Hu's (critical path) algorithm and s^* is an optimal schedule for the makespan problem. For $m \geq 3$ this bound is only asymptotically achievable. In Section 3 we tighten (1) for $m \geq 3$. Moreover, in this section we also present a bound that establishes the relationship between the deviation of the criterion value from its optimum and the parameters characterizing the problem. Providing an insight into the nature of Hu's algorithm these performance guarantees are also useful in the development of various algorithms based on the calculation of lower bounds for the optimal value of the criterion.

A distinctive feature of both bounds presented in Section 3 is that they are achievable for arbitrary large instances of the problem.

In contrast to the makespan problem much less is known on the worst-case performance of approximation algorithms for the maximum lateness problem. Since the latter problem is a straightforward generalization of the former, the establishment of such results is viewed as a logical step in theoretical study in the field. Thus, the following performance guarantee has been derived in (Zinder and Roper 1998) for the algorithm introduced earlier by the first of the author:

$$L_{max}(s^Z) \leq \left(2 - \frac{2}{m}\right) L_{max}(s^*) + \left(1 - \frac{2}{m}\right) \max_{a \in N} d_a - r(m), \tag{2}$$

where s^Z is a schedule constructed by the approximation algorithm, s^* is an optimal schedule for the maximum lateness problem, and

$$r(m) = \begin{cases} \frac{m-3}{m} & \text{for } m \text{ odd} \\ \frac{m-2}{m} & \text{for } m \text{ even.} \end{cases}$$

If all due dates are equal to zero, (2) converts into the bound presented in (Braschi and Trystram 1994) for the Coffman-Graham algorithm (Coffman and Graham 1972) for the makespan problem. On the other hand, P. Brucker, M.R. Garey and D.S. Johnson (Brucker et al. 1977) proved that a straightforward extension of Hu's algorithm yields an optimal schedule for the maximum lateness problem if the graph corresponding to the partially ordered set of operations is an in-tree. In the light of these results, one may anticipate the following bound for the Brucker-Garey-Johnson algorithm

$$L_{max}(s^{BGJ}) \leq \left(2 - \frac{1}{m-1}\right) L_{max}(s^*) + \left(1 - \frac{1}{m-1}\right) \max_{a \in N} d_a, \tag{3}$$

where s^{BGJ} is a schedule constructed by the Brucker-Garey-Johnson algorithm and s^* is an optimal schedule for the maximum lateness problem. In order to show that this conjecture is incorrect, consider the graph depicted in Figure 1. This graph is comprised of k identical sections and one terminal section, containing m rows each. Last (bottom) row of each identical section contains only one node, whereas all other rows in the graph contain $m+1$ nodes each. The shaded nodes precede all nodes in the next row, whereas each non-shaded node has no successors. Each operation corresponding to a node in the first (top) row has a due date of one unit of time. Each operation from any other row has a due date of the previous row plus one. To construct an optimal schedule s^* we consider a list in which operations are arranged in increasing order of their due dates, and the operations corresponding to the shaded nodes precede all operations corresponding to the non-shaded nodes with the same due date. Because all operations j in the same row have the same due dates, there exists a schedule s^{BGJ} in which the operation corresponding to the shaded node of each row is processed after the operations corresponding to non-shaded nodes of this row. The corresponding optimal schedule and the schedule s^{BGJ} constructed by the Brucker-Garey-Johnson algorithm are presented in Figure 2 and Figure 3, respectively. It is easy to see that $\max_{j\in N} d_j = m(k+1)$, $L_{max}(s^{BGJ}) = mk - k + m$, $L_{max}(s^*) = 1$. Therefore, (3) does not hold if $k \geq m-2$. A tight performance guarantee for the Brucker-Garey-Johnson algorithm is derived in Section 2. It is shown that this bound is achievable for arbitrary large instances of the problem.

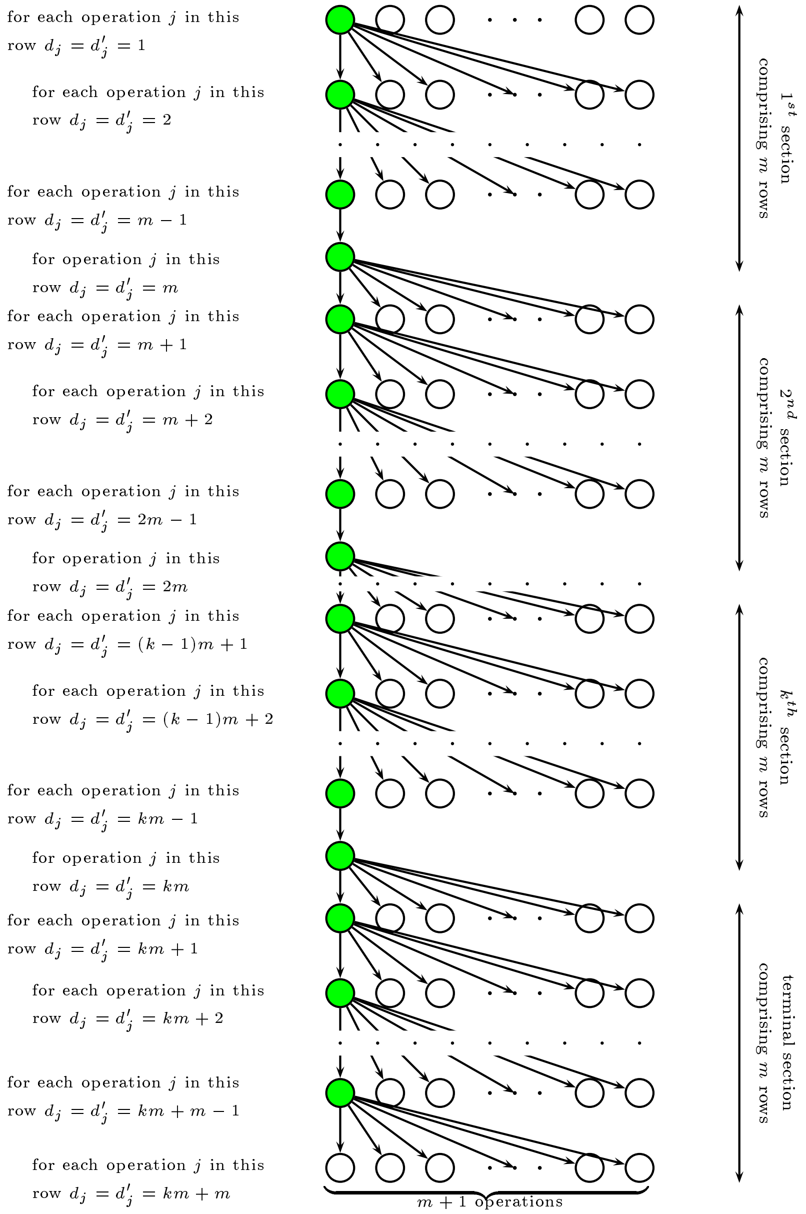

Figure 1: The partially ordered set of tasks considered in the counterexample to the conjecture (3) and in Theorem 1

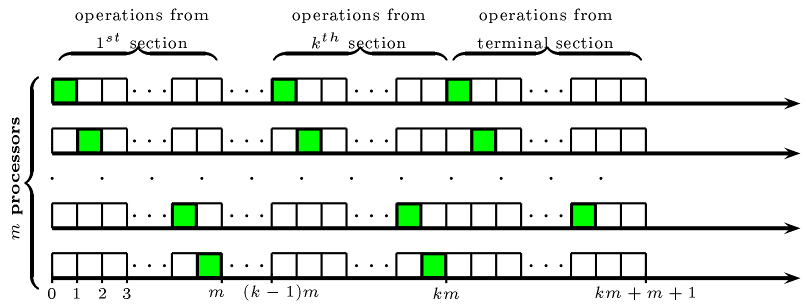

Figure 2: An optimal schedule for the maximum lateness problem

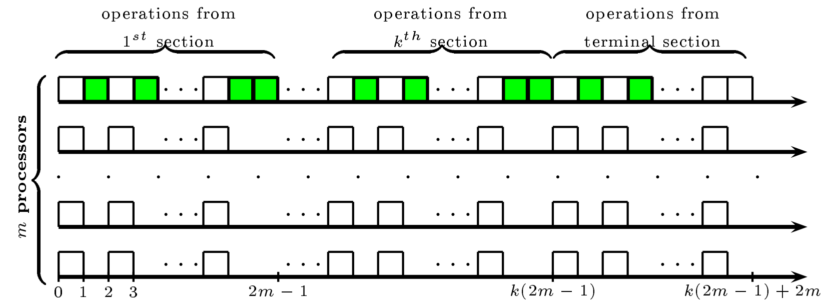

Figure 3: A non-optimal schedule constructed by the Brucker-Garey-Johnson algorithm

2 The worst-case performance of the Brucker-Garey-Johnson algorithm.

Both Hu's algorithm and the Brucker-Garey-Johnson algorithm can be viewed as two-phase procedures. The first phase of each algorithm associates with each operation a number which indicates the priority (urgency) of the operation. The second phase forms a list in which the operations are arranged in accord with their priorities and constructs a schedule using the following algorithm.

List Algorithm

Consider sequentially the time points $t = 0$, $t = 1$, $t = 2$, and so on. For each such a time point, starting from $t = 0$, the list is scanned from left to right and the first encountered operation available for processing is assigned to an idle processor. At each time point the algorithm continues to scan the list and allocate available operations for processing until at least one of the following events occurs: (a) the end of the list has been reached; or (b) all processors have had operations assigned to them. Then the allocation of the remaining operations is made for subsequent time points until all operations have been assigned.

The priority associated with each operation by the Brucker-Garey-Johnson algorithm is called a modified due date, and the operations are arranged in the list in nondecreasing order of their modified due dates. We will denote the modified due date associated with operation

a by d'_a. Modified due-date d'_a depends on the modified due dates of the immediate successors of operation a. Operation b is an immediate successor of operation a if $(a, b) \in \alpha$ and there is no operation q such that $(a, q) \in \alpha$ and $(q, b) \in \alpha$. We will denote the set of all immediate successors of operation a by $K(a)$. In (Brucker et al. 1977), the algorithm computing the modified due dates was given only for in-trees. The algorithm below is a straightforward extension of this algorithm to the case of an arbitrary partially ordered set.

1. For each operation a that do not have successors, set $d'_a = d_a$.

2. Select an operation a which has not been assigned its modified due date d'_a and whose all immediate successors have been assigned their modified due dates. If no such an operation exists, halt with all modified due dates assigned.

3. Set $d'_a = \min\{d_a, \min_{b \in K(a)} (d'_b - 1)\}$ and return to step 2.

Lemma 1 *For any schedule s*

$$\max_{q \in N}(C_q(s) - d'_q) = L_{max}(s). \tag{4}$$

Proof. Since $d'_q \leq d_q$ for any operation q, we have

$$\max_{q \in N}(C_q(s) - d'_q) \geq \max_{q \in N}(C_q(s) - d_q) = L_{max}(s).$$

Among all operations a satisfying

$$\max_{q \in N}(C_q(s) - d'_q) = C_a(s) - d'_a$$

select an operation with the largest completion time. Let it be operation b. If $d_b = d'_b$, then

$$\max_{q \in N}(C_q(s) - d'_q) = C_b(s) - d_b \leq L_{max}(s)$$

and (4) holds. Suppose that $d_b > d'_b$, then there is an immediate successor of operation b, say operation v, such that $d'_b = d'_v - 1$. Since $C_v(s) \geq C_b(s) + 1$, we have

$$C_b(s) - d'_b = C_b(s) - (d'_v - 1) = (C_b(s) + 1) - d'_v \leq C_v(s) - d'_v,$$

which contradicts the selection of operation b. $\square$

Let s be an arbitrary schedule on (N, α), which schedules operations from $t = 0$. Following (Zinder and Roper 1998) we will describe this schedule by means of $m \times \max_{v \in N} C_v(s)$ matrix.

Each column of this matrix corresponds to a unit time period and each row corresponds to a processor. The entry at the intersection of row k and column t is zero if the processor k is idle during the time period $[t-1,t]$, and $a \in N$, if the processor k processes operation a during this period. We will refer to this matrix as a schedule matrix and denote it $A(s)$.

Let s^{BGJ} be a schedule constructed by the Brucker-Garey-Johnson algorithm. From the set of all operations j such that

$$C_j(s^{BGJ}) - d'_j = L_{max}(s^{BGJ})$$

choose an operation p with the smallest completion time. Let $M(p)$ be the set of all operations $a \in N$ such that $d'_a \leq d'_p$. It is easy to see that for any $a \in M(p)$ inequality $C_a(s^{BGJ}) \leq C_p(s^{BGJ})$ holds, and that any predecessor of an operation from $M(p)$ also belongs to $M(p)$.

Lemma 2 *In the schedule matrix $A(s^{BGJ})$ the column $C_p(s^{BGJ})-1$ contains m operations from $M(p)$.*

Proof. Suppose that the column $C_p(s^{BGJ})-1$ conatins less than m operations from $M(p)$. Because the operations are arranged in the list in nondecreasing order of their modified due dates and because in column $C_p(s^{BGJ})-1$ at least one processor is either idle or processes an operation with the modified due date greater than d'_p, operation p is unavailable for processing at time $C_p(s^{BGJ})-1$. This means that there exists an operation q such that $C_q(s^{BGJ}) = C_p(s^{BGJ})-1$ and $(q,p) \in \alpha$. Hence $d'_q \leq d'_p - 1$, and

$$C_q(s^{BGJ}) - d'_q \geq C_q(s^{BGJ}) - (d'_p - 1) = (C_q(s^{BGJ}) + 1) - d'_p = C_p(s^{BGJ}) - d'_p,$$

which contradicts the selection of operation p. □

Theorem 1 *If s^* is an optimal schedule for the maximum lateness problem, then*

$$L_{max}(s^{BGJ}) \leq \left(2 - \frac{1}{m}\right) L_{max}(s^*) + \left(1 - \frac{1}{m}\right) \max_{j \in N} d_j - \left(\frac{m-1}{m}\right). \tag{5}$$

For any positive integer $\hat{n}$ there is an instance of the maximum lateness problem with $n \geq \hat{n}$ such that (5) is an equality.

Proof. If $C_p(s^{BGJ}) = 1$, then

$$L_{max}(s^{BGJ}) \leq L_{max}(s^{BGJ}) + \frac{m-1}{m}(\max_{j \in N} d_j - d'_p) + \frac{m-1}{m}(C_p(s^{BGJ}) - 1)$$

$$= L_{max}(s^{BGJ}) + \frac{m-1}{m}(C_p(s^{BGJ}) - d'_p) + \frac{m-1}{m} \max_{j\in N} d_j - \frac{m-1}{m}$$

and since $L_{max}(s^{BGJ}) = C_p(s^{BGJ}) - d'_p$ and s^{BGJ} is optimal

$$= \left(2 - \frac{1}{m}\right) L_{max}(s^*) + \left(1 - \frac{1}{m}\right) \max_{j\in N} d_j - \left(\frac{m-1}{m}\right).$$

Suppose that $C_p(s^{BGJ}) > 1$. Let w be the number of all columns t each containing less than m operations from $M(p)$ and such that $t < C_p(s^{BGJ})$, c be the number of all columns t each containing m operations from $M(p)$ and such that $t < C_p(s^{BGJ})$, and U be the set comprising operation p and all operations from the column $C_p(s^{BGJ}) - 1$. Select an arbitrary operation from this set and denote this operation by b_0. Among all columns t containing less than m operations from $M(p)$ and such that $t < C_p(s^{BGJ})$ select the column with the largest value of t. Let it be column t'. Because the operations are arranged in the list in nondecreasing order of their modified due dates and because in column t' at least one processor is either idle or processes an operation with the modified due date greater than d'_p, operation b_0 is unavailable for processing on the interval $[t'-1, t']$. This means that there exists an operation b_1 such that $C_{b_1}(s^{BGJ}) \geq t'$ and $(b_1, b_0) \in \alpha$. Since $d'_{b_1} < d'_{b_0} \leq d'_p$, we have $b_1 \in M(p)$. Now among all columns t containing less than m operations from $M(p)$ and such that $t < C_{b_1}(s^{BGJ})$ select the column with the largest value of t. Let it be column t''. Using the same line of reasoning we conclude that since $b_1 \in M(p)$ and the column t'' contains less than m operations from $M(p)$, there must exist an operation b_2 such that $C_{b_2}(s^{BGJ}) \geq t''$ and $(b_2, b_1) \in \alpha$. Proceeding leftward in the matrix $A(s^{BGJ})$ we obtain a sequence of operations $b_0, \ldots, b_w$ such that $b_i \in M(p)$, for all i, and $(b_{i+1}, b_i) \in \alpha$, for all $i < w$. Hence, for any schedule s and any operation $b \in U$, $C_b(s) \geq w+1$. According to Lemma 2 $|U| = m+1$. Therefore, $\max_{j\in M(p)} C_j(s^*) \geq w+2$. Using this inequality together with the inequality

$$\max_{j\in M(p)} C_j(s^*) \geq \frac{|M(p)|}{m} \geq \frac{mc+w+1}{m},$$

we obtain

$$C_p(s^{BGJ}) = c + w + 1 = \frac{mc+w+1}{m} + (w+2)\left(\frac{m-1}{m}\right) - \left(\frac{m-1}{m}\right)$$
$$\leq \left(2 - \frac{1}{m}\right) \max_{q\in M(p)} C_q(s^*) - \left(1 - \frac{1}{m}\right).$$

Hence,

$$L_{max}(s^{BGJ}) = C_p(s^{BGJ}) - d'_p \leq \left(2 - \frac{1}{m}\right) \max_{j\in M(p)} C_j(s^*) - \left(\frac{m-1}{m}\right) - d'_p$$

$$= \max_{j\in M(p)}\{\left(2-\frac{1}{m}\right)C_j(s^*)-d'_p\} - \left(\frac{m-1}{m}\right)$$

and since $d'_j \le d'_p$ for any $j \in M(p)$

$$\le \max_{j\in M(p)}\{\left(2-\frac{1}{m}\right)C_j(s^*)-d'_j\} - \left(\frac{m-1}{m}\right) \le \max_{j\in N}\{\left(2-\frac{1}{m}\right)C_j(s^*)+\max_{j\in N} d_j - d'_j\}$$
$$-\left(\frac{m-1}{m}\right) - \max_{j\in N} d_j \le \max_{j\in N}\{\left(2-\frac{1}{m}\right)C_j(s^*)+\left(2-\frac{1}{m}\right)(\max_{j\in N} d_j - d'_j)\} - \left(\frac{m-1}{m}\right)$$
$$-\max_{j\in N} d_j = \left(2-\frac{1}{m}\right)\max_{j\in N}(C_j(s^*)-d'_j) + \left(1-\frac{1}{m}\right)\max_{j\in N} d_j - \left(\frac{m-1}{m}\right)$$
$$= \left(2-\frac{1}{m}\right)L_{max}(s^*) + \left(1-\frac{1}{m}\right)\max_{j\in N} d_j - \left(\frac{m-1}{m}\right).$$

The tightness of (5) follows from the example presented in Figures 1, 2, and 3. Indeed, in this example (5) is equality for any k and m. □

3 The Worst-Case Performance of Hu's Algorithm

Let l_a be the length of the longest path emanating from the node associated with operation a in the acyclic directed graph corresponding to the partially ordered set of operations. If $K(a) = \emptyset$, then $l_a = 0$, and if $K(a) \neq \emptyset$, then $l_a = \max_{b\in K(a)} l_b + 1$. According to Hu's algorithm the operations are arranged in a list in nonincreasing order of l_a values and then are scheduled in accord with the list algorithm. On the other hand, if all due dates are equal to zero, the maximum lateness problem converts into the makespan problem. Moreover, in this case $-d'_a = l_a$, for all operations a, and therefore, the Brucker-Garey-Johnson algorithm is equivalent to Hu's algorithm. In what follows we assume that all due dates are equal to zero and replace $L_{max}(s)$ by $C_{max}(s)$ and s^{BGJ} by s^{CP} emphasizing that this schedule is based on the idea of the critical path method. Certainly, in the case of zero due dates the selection of task p and the corresponding results presented in the previous section hold. In particular,

$$C_p(s^{CP}) + l_p = \max_{a\in N}(C_a(s^{CP}) + l_a) = C_{max}(s^{CP})$$

and (5) gives

$$C_{max}(s^{CP}) \le \left(2-\frac{1}{m}\right)C_{max}(s^*) - \left(\frac{m-1}{m}\right), \tag{6}$$

where s^* is an optimal schedule for the makespan problem. It is not difficult to see from Figures 5 and 6 that for the partially ordered set of operations presented in Figure 4

$$C_{max}(s^{CP}) = 2m \quad \text{and} \quad C_{max}(s^*) = m + 1,$$

and therefore, for this instance of the makespan problem (6) is an equality.

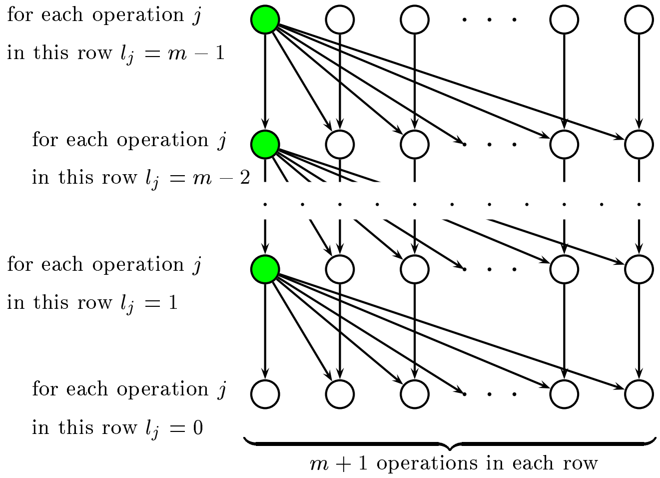

Figure 4: The partially ordered set which shows the tightness of bound (6)

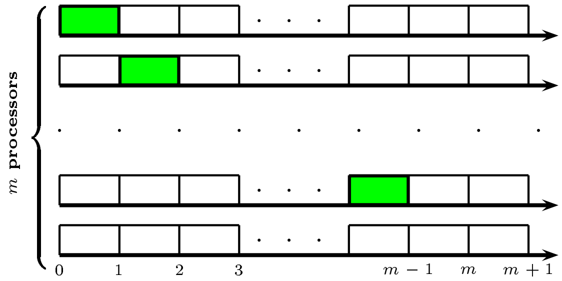

Figure 5: An optimal schedule for the makespan problem

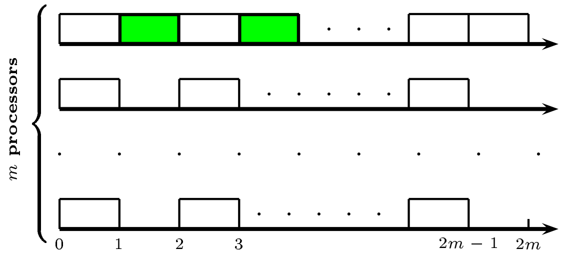

Figure 6: A non-optimal schedule constructed by Hu's algorithm

For the purpose of further characterization of the worst-case performance of Hu's algorithm we need to examine the structure of schedule s^{CP} in more detail. In doing so we will define a sequence of numbers $\beta_1, \ldots, \beta_{C_p(s^{CP})}$ such that $\beta_1 \geq \ldots \geq \beta_{C_p(s^{CP})}$ and the corresponding sets of operations $S_1, \ldots, S_{C_p(s^{CP})}$, where $S_t = \{a : C_a(s^{CP}) = t \;\text{ and }\; l_a \geq \beta_t\}$, in the following way. Let $\beta_{C_p(s^{CP})} = l_p$. Then it is easy to see that $p \in S_{C_p(s^{CP})}$ and $l_a = l_p$ for any operation $a \in S_{C_p(s^{CP})}$. Suppose that β_t and $S_t \neq \emptyset$ have been defined for some $t > 1$. If the column $t-1$ contains m operations a satisfying $l_a \geq \beta_t$, then $\beta_{t-1} = \beta_t$. Note that in this case S_{t-1} coincides with the set of all operations in the column $t-1$. Suppose that on the time interval $[t-2, t-1]$ at least one processor is either idle or processes an operation q with $l_q < \beta_t$. Then any a satisfying $C_a(s^{CP}) > t-1$ and $l_a \geq \beta_t$ has a predecessor in the column $t-1$. In order to prove this assertion we assume that at least one operation a satisfying $C_a(s^{CP}) > t-1$ and $l_a \geq \beta_t$ does not have a predecessor in the column $t-1$. Among all these operations select an operation a' with the smallest completion time. Since the operations are arranged in the list in the nonincreasing order of their l_a values, a' is not available for processing at time $t-2$, i.e. there is an operation a'' such that $(a'', a') \in \alpha$ and $C_{a''}(s^{CP}) \geq t-1$. Because a' does not have a predecessor in the column $t-1$, $C_{a''}(s^{CP}) > t-1$ and a'' also does not have a predecessor in this column. Since $l_{a''} > l_{a'} \geq \beta_t$ the existence of a'' contradicts the selection of a'. Therefore, each operation a satisfying $C_a(s^{CP}) > t-1$ and $l_a \geq \beta_t$ has a predecessor in the column $t-1$. Let S' be the set of all these predecessors, then

$$\beta_{t-1} = \min_{q \in S'} l_q.$$

Denote the set $\cup_{t=1}^{C_p(s^{CP})} S_t$ by S and define for each $q \in S$ a set $V(q)$ as follows. If $l_q = l_p$, then $V(q) = \emptyset$. Suppose that $l_q > l_p$. Then $V(q) = \{v_1, \ldots, v_{\lambda(q)}\}$, where $\lambda(q) = l_q - l_p$, $(q, v_1) \in \alpha$, $l_{v_{\lambda(q)}} = l_p$, and $(v_{i-1}, v_i) \in \alpha$, for all $i > 1$. Note that for some q there may be more than one way of selecting the set $V(q)$. In this case we select one of these possible sets at random. Let $M = S \cup (\cup_{q \in S} V(q))$, then it is easy to see that $C_a(s^{CP}) \leq C_p(s^{CP})$ for each $a \in M$.

Lemma 3 *If $v \in M$ and $v \notin S$, then $\beta_{C_v(s^{CP})} = l_v + 1$.*

Proof. Suppose that this statement does not hold for some v. Since $v \notin S$, $v = v_{i^*}$ for some $i^* \leq \lambda(q)$ and some $V(q) = \{v_1, \ldots, v_{\lambda(q)}\}$. Denote the operation q by v_0. Then

$$\beta_{C_{v_{i^*-1}}(s^{CP})} \geq \beta_{C_{v_{i^*}}(s^{CP})} \geq l_{v_{i^*}} + 2 = (l_{v_{i^*}} + 1) + 1 = l_{v_{i^*-1}} + 1.$$

Hence, $v_{i^*-1} \notin S_{C_{v_{i^*-1}}(s^{CP})}$ and from the definition of β's and inequality above we have

$$\beta_{C_{v_{i^*-1}}(s^{CP})} \geq \beta_{C_{v_{i^*}}(s^{CP})} + 1 \geq l_{v_{i^*-1}} + 2.$$

Thus, $l_{v_i} + 2 \leq \beta_{C_{v_i}(s^{CP})}$ for all $0 \leq i < i^*$, which contradicts the assumption that $q \in S$. □

Lemma 4 *Let $a \in (V(q) - S)$ for some $q \in S$ and let b be the operation with the largest completion time among all operations $x \in V(q) \cup \{q\}$ satisfying $(x, a) \in \alpha$ and $x \in S$. Then $|S_{C_b(s^{CP})}| = m$ and any column t, where $C_b(s^{CP}) < t < C_a(s^{CP})$ and $|S_t| < m$, contains an operation from $V(q) - S$.*

Proof. Suppose that this statement does not hold for some t', where $C_b(s^{CP}) < t' < C_a(s^{CP})$ and $|S_{t'}| < m$. Because column t' does not contain any operation from $V(q) - S$, from the selection of b it follows the existence of operation $v' \in V(q) \cup \{q\}$ and operation $v'' \in V(q)$ such that $C_{v'}(s^{CP}) < t' < C_{v''}(s^{CP})$ and $l_{v'} = l_{v''} + 1$. Since $|S_{t'}| < m$, by Lemma 3 and the definition of β's we have

$$\beta_{C_{v'}(s^{CP})} \geq \beta_{t'} \geq \beta_{C_{v''}(s^{CP})} + 1 = (l_{v''} + 1) + 1 = l_{v'} + 1.$$

Hence, $v' \notin S$ and from inequality above

$$\beta_{C_{v'}(s^{CP})} \geq \beta_{t'} + 1 \geq (l_{v'} + 1) + 1 = l_{v'} + 2$$

which contradicts Lemma 3.

Now suppose that $|S_{C_b(s^{CP})}| < m$. Consider the operation $v \in V(q)$ such that $l_b = l_v + 1$. From the selection of b it follows that $v \notin S$, and therefore by Lemma 3 $\beta_{C_b(s^{CP})} \geq \beta_{C_v(s^{CP})} + 1 = (l_v + 1) + 1 = l_b + 1$, which contradicts the assumption that $b \in S$. $\square$

Lemma 5 *If a is the only operation from M in the column $C_a(s^{CP})$, then $(a, b) \in \alpha$ for any operation $b \in M$ such that $C_b(s^{CP}) > C_a(s^{CP})$.*

Proof. Suppose that there exists an operation $b \in M$ such that $C_b(s^{CP}) > C_a(s^{CP})$ and $(a, b) \notin \alpha$. Among all such operations select an operation, say u, with the smallest completion time. By the definition of S each column t, where $t \leq C_p(s^{CP})$, contains at least one operation from S. Therefore $a \in S$. Because a is the only operation from S in the column $C_a(s^{CP})$, $(a, x) \in \alpha$ for each $x \in S$ satisfying the inequality $C_x(s^{CP}) > C_a(s^{CP})$. Hence $u \notin S$ and $u \in V(q)$ for some $q \in S$. Let y be the operation with the largest completion time among all operations $x \in V(q) \cup \{q\}$ such that $(x, u) \in \alpha$ and $x \in S$. By the selection of u and because the column $C_a(s^{CP})$ contains only one operation from M, $C_y(s^{CP}) < C_a(s^{CP})$. Then by Lemma 4 the column $C_a(s^{CP})$ must contain an operation from $V(q) - S$ which contradicts the assumption that this column contains only one operation from M. $\square$

Now we will split M into disjoint subsets $M_1, \ldots, M_h$ using the following procedure. Suppose that there exists a column t such that this column contains only one operation from M, say operation a, and either $C_a(s^{CP}) = 1$ or $C_a(s^{CP}) > 1$ and $(x, a) \in \alpha$ for all $x \in M$ satisfying the inequality $C_x(s^{CP}) < C_a(s^{CP})$. Among all such operations a select the operation with the smallest completion time. Let it be operation a_1. If such an operation does not exist, select operation p as a_1. In both cases $M_1 = \{x : x \in M \text{ and } C_x(s^{CP}) \leq C_{a_1}(s^{CP})\}$. Suppose that the sets $M_1, \ldots, M_k$ and the corresponding operations $a_1, \ldots, a_k$ have been defined. If

$a_k = p$, then the procedure terminates. Suppose that $a_k \neq p$ and there exists a column t such that this column contains only one operation from M, say operation a, and either $C_a(s^{CP}) = C_{a_k}(s^{CP})+1$ or $C_a(s^{CP}) > C_{a_k}(s^{CP})+1$ and $(x,a) \in \alpha$ for all $x \in M$ satisfying the inequality $C_x(s^{CP}) < C_a(s^{CP})$. Among all such operations a select the operation with the smallest completion time. Let it be operation a_{k+1}. If such operation a does not exist, then select as a_{k+1} the operation p. In both cases $M_{k+1} = \{x : x \in M \text{ and } C_{a_k}(s^{CP}) < C_x(s^{CP}) \leq C_{a_{k+1}}(s^{CP})\}$.

Let $\tau_0 = 0$ and $\tau_i = C_{a_i}(s^{CP})$ for all $1 \leq i \leq h$.

Lemma 6 *If column t', where $\tau_i < t' < \tau_{i+1}$ and $0 \leq i < h$, contains only one operation from M, then there exists a column t'' such that $\tau_i < t'' < t'$ and $|S_{t''}| = m$. Moreover, either $t'' = t' - 1$ or each column t, where $t'' < t < t'$, contains at least two operations from M.*

Proof. Let a be the operation from M which is processed on the time interval $[t'-1, t']$. By the construction of M_i there exists an operation $x \in M$ such that $(x,a) \notin \alpha$ and $C_x(s^{CP}) < C_a(s^{CP})$. Among all these operations x select an operation with the largest completion time. Let it be operation v. If $l_v < l_a$, then $\beta_{C_v(s^{CP})} \geq \beta_{t'} = l_a > l_v$. Hence, $v \notin S$ and $\beta_{C_v(s^{CP})} \geq \beta_{t'} + 1 > l_v + 1$, which contradicts Lemma 3. Therefore, $l_v \geq l_a$.

Since $K(a) \neq \emptyset$, there exists an operation b such that $(v,b) \in \alpha$ and $l_v = l_b + 1$. It follows from the selection of v that $C_b(s^{CP}) \geq C_a(s^{CP})$. If $C_b(s^{CP}) = C_a(s^{CP})$, then since $S_{t'} = \{a\}$, $l_a = \beta_{t'} > l_b$. Suppose that $C_b(s^{CP}) > C_a(s^{CP})$. If $b \in S$, then by the construction of S we have $(a,b) \in \alpha$ and hence $l_a > l_b$. If $b \notin S$, then there exists an operation $y \in S$ such that $C_y(s^{CP}) > C_a(s^{CP})$ and $l_y > l_b$. Because $(a,y) \in \alpha$, $l_a > l_y > l_b$. Since in all these cases $l_a > l_b$, we have $l_a \geq l_b + 1 = l_v$ which together with the previously obtained inequality $l_v \geq l_a$ gives $l_v = l_a$.

Let $C_v(s^{CP}) < C_a(s^{CP}) - 1$ and let t be any positive integer satisfying inequalities $C_v(s^{CP}) < t < C_a(s^{CP})$. If $|S_t| < m$, then $\beta_{C_v(s^{CP})} \geq \beta_t > \beta_{C_a(s^{CP})} = l_a = l_v$. Hence, $v \notin S$ and $\beta_{C_v(s^{CP})} \geq \beta_t + 1 \geq \beta_{C_a(s^{CP})} + 2 = l_a + 2 = l_v + 2$, which contradicts Lemma 3. Therefore, in this case any column t, where $C_v(s^{CP}) < t < C_a(s^{CP})$, can be selected as t''.

Suppose that $C_v(s^{CP}) = C_a(s^{CP}) - 1$. If $v \in S$, then $l_v \geq \beta_{C_v(s^{CP})} \geq \beta_{t'} = l_a$, which by virtue of the equality $l_a = l_v$ implies that $\beta_{C_v(s^{CP})} = \beta_{t'}$. Hence, $|S_{C_v(s^{CP})}| = m$ and $C_v(s^{CP})$ can be chosen as t''. Suppose that $v \notin S$. Then $v \in V(q)$ for some $q \in S$. Among all operations $x \in V(q) \cup \{q\}$ such that $x \in S$ and $(x,v) \in \alpha$ select the operation b with the largest completion time. Then by Lemma 4 $|S_{C_b(s^{CP})}| = m$ and all columns between column

$C_b(s^{CP})$ and $C_v(s^{CP})$ contain at least two operations from M. Therefore, in this case $C_b(s^{CP})$ can be chosen as t''.□

For any schedule s and any i such that $1 \leq i \leq h$, let

$$C^i_{max}(s) = \max_{a \in M_i} C_a(s) - \min_{a \in M_i} C_a(s) + 1.$$

Consider schedule s^{CP}. Then the total number of columns t such that $\tau_{i-1} < t \leq \tau_i$ is equal to $c_i + w_{i1} + w_{i2}$, where c_i is the number of columns containing exactly m operations from M_i, w_{i1} is the number of columns containing only one operation from M_i, and w_{i2} is the number of columns containing more than one and less than m operations from M_i. In particular, using this notation we have

$$|M_i| \geq mc_i + w_{i1} + 2w_{i2}. \tag{7}$$

Lemma 7 *For any $m \geq 3$ and any i such that $1 \leq i \leq h$*

$$C^i_{max}(s^{CP}) - C^i_{max}(s^*) \leq \frac{|M_i|}{m(m-1)} + \frac{(m-3)(w_{i1} + w_{i2})}{m-1}. \tag{8}$$

Proof. Suppose that $i < h$, then for any x such that $x \neq a_i$ and $x \in M_i$ we have $(x, a_i) \in \alpha$. Therefore,

$$C^i_{max}(s^*) \geq \frac{mc_i + w_{i1} + 2w_{i2} - 1}{m} + 1, \tag{9}$$

where s^* is an optimal schedule for the makespan problem. Using elementary algebra it is easy to see that

$$C^i_{max}(s^{CP}) = c_i + w_{i1} + w_{i2} = \frac{mc_i + w_{i1} + 2w_{i2} - 1}{m} + 1 + \frac{mc_i + w_{i1} + 2w_{i2}}{m(m-1)}$$

$$+ \frac{(m-3)(w_{i1} + w_{i2})}{m-1} + \frac{w_{i1} - c_i + 2 - m}{m-1} - \frac{1}{m(m-1)}$$

and using (7) and (9)

$$\leq C^i_{max}(s^*) + \frac{|M_i|}{m(m-1)} + \frac{(m-3)(w_{i1} + w_{i2})}{m-1} + \frac{w_{i1} - c_i + 2 - m}{m-1} - \frac{1}{m(m-1)}.$$

By Lemma 6 $w_{i1} - c_i \leq 1$, and since $m \geq 3$, we obtain

$$C^i_{max}(s^{CP}) - C^i_{max}(s^*) \leq \frac{|M_i|}{m(m-1)} + \frac{(m-3)(w_{i1} + w_{i2})}{m-1}.$$

It is easy to see that

$$C^h_{max}(s^*) \geq \frac{mc_h + w_{h1} + 2w_{h2}}{m},$$

and therefore,

$$C^h_{max}(s^{CP}) = c_h + w_{h1} + w_{h2} = \frac{mc_h + w_{h1} + 2w_{h2}}{m} + \frac{mc_h + w_{h1} + 2w_{h2}}{m(m-1)}$$

$$+\frac{(m-3)(w_{h1}+w_{h2})}{m-1} + \frac{w_{h1}-c_h}{m-1} \le C^h_{max}(s^*) + \frac{|M_h|}{m(m-1)} + \frac{(m-3)(w_{h1}+w_{h2})}{m-1} + \frac{w_{h1}-c_h}{m-1}.$$

Lemma 2 and Lemma 6 imply that $w_{h1} - c_h \le 0$, and therefore, (8) holds also for $i = h$. □

Theorem 2 *If s^* is a schedule, which is optimal for the criterion $C_{max}(s)$, then for any $m \ge 3$*

$$C_{max}(s^{CP}) - C_{max}(s^*) \le \frac{n}{m(m-1)} + \frac{l(m-3)}{m-1}, \tag{10}$$

where n is the number of operations and l is the length of the longest path in the corresponding graph. For any positive integer $\hat{n}$ there is an instance of the makespan problem with $n \ge \hat{n}$ such that (10) is an equality.

Proof. We have $C_{max}(s^{CP}) = \sum_{i=1}^{h} C^i_{max}(s^{CP}) + l_p$. On the other hand, from Lemma 5 and the construction of the sets M_i it follows that in any schedule operations from M_{i+1} are processed only after the completion of all operations from M_i. Hence, $C_{max}(s^*) \ge \sum_{i=1}^{h} C^i_{max}(s^*) + l_p$. Therefore using Lemma 7 we have

$$C_{max}(s^{CP}) - C_{max}(s^*) \le \sum_{i=1}^{h}[C^i_{max}(s^{CP}) - C^i_{max}(s^*)] \le \frac{\sum_{i=1}^{h}|M_i|}{m(m-1)} + \frac{(m-3)\sum_{i=1}^{h}(w_{i1}+w_{i2})}{m-1}.$$

From the definition of S it follows that $l \ge \sum_{i=1}^{h}(w_{i1}+w_{i2})$. Hence,

$$C_{max}(s^{CP}) - C_{max}(s^*) \le \frac{n}{m(m-1)} + \frac{(m-3)l}{m-1}.$$

In order to show that (10) is tight, we consider the partially ordered set of operations corresponding to the graph in Figure 7. This graph is comprised of an initial section followed by k identical sections and one terminal section. The initial section is comprised of m rows, whereas each of k identical sections and the terminal section is comprised of $m-1$ rows. The last (bottom) row of the initial section and each bottom row of each identical section contains two nodes. All other rows in the graph contain $m+1$ nodes each. Each shaded node precedes all nodes in the next row, whereas each non-shaded node precedes exactly one non-shaded node in the next row. To construct an optimal schedule s^* we consider a list in which operations are arranged in decreasing order of their l_j values, and each operation corresponding to the shaded node precedes all operations corresponding to the non-shaded nodes with the same l_j value. The schedule s^* is given in Figure 8. Because all operations j in the same row have the same l_j value, there exists a schedule s^{CP} in which each operation corresponding to the shaded node of each row, containing more than m operations, is processed after the operations corresponding to the non-shaded nodes of this row. This schedule is depicted in Figure 9. Since $C_{max}(s^*) = 2m + k(m-1)$, $C_{max}(s^{CP}) = 4m - 3 + k(2m-3)$, $n = 2m^2 + km(m-1)$ and $l = 2m - 1 + k(m-1)$, (10) becomes an equality. $\square$

Theorem 3 *If s^* is a schedule, which is optimal for the criterion $C_{max}(s)$, then for any $m \geq 3$*

$$C_{max}(s^{CP}) \leq \left(2 - \frac{1}{m-1}\right) C_{max}(s^*) - \left(\frac{m-3}{m-1}\right). \tag{11}$$

For any positive integer $\hat{n}$ there is an instance of the makespan problem with $n \geq \hat{n}$ such that (11) is an equality.

Proof. It is easy to see that for all i such that $1 \leq i \leq h$

$$C^i_{max}(s^*) \geq \frac{|M_i|}{m}. \tag{12}$$

Suppose that $i < h$. From the construction of the set S it follows that

$$C^i_{max}(s^*) \geq w_{i1} + w_{i2}. \tag{13}$$

Using (12) and (13) together with Lemma 7 we obtain

$$C^i_{max}(s^{CP}) \leq \left(2 - \frac{1}{m-1}\right) C^i_{max}(s^*).$$

Let U be the set of all operations from M_h which are processed on the time interval $[C_p(s^{CP}) - 2, C_p(s^{CP})]$. Lemma 2 implies that $U \subseteq S$ and $|U| \geq m+1$. Let w be the number of columns t such that $\tau_{h-1} < t < \tau_h$ and column t contains less than m operations from M. Let they be columns $t_1, \ldots, t_w$, where $t_i < t_{i+1}$ for each $i < w$. Let x be an arbitrary operation from U. Since $x \in S$ there is an operation say, $x_w \in S_{t_w}$, such that $(x_w, x) \in \alpha$. Since $x_w \in S$, there is an operation say, $x_{w-1} \in S_{t_{w-1}}$, such that $(x_{w-1}, x_w) \in \alpha$. Continuing this reasoning we conclude that there is a chain of w operations in M_h which precedes any operation from U. Since $w \geq w_{h1} + w_{h2} - 1$ and $|U| \geq m+1$, we obtain $C^h_{max}(s^*) \geq w_{h1} + w_{h2} + 1$, which together with Lemma 7 gives

$$C^h_{max}(s^{CP}) \leq \left(2 - \frac{1}{m-1}\right) C^h_{max}(s^*) - \left(\frac{m-3}{m-1}\right).$$

Since $\sum_{i=1}^{h} C^i_{max}(s^*) + l_p \le C_{max}(s^*)$, we have

$$C_{max}(s^{CP}) = \sum_{i=1}^{h-1} C^i_{max}(s^{CP}) + C^h_{max}(s^{CP}) + l_p$$

$$\le \left(2 - \frac{1}{m-1}\right) \sum_{i=1}^{h-1} C^i_{max}(s^*) + \left(2 - \frac{1}{m-1}\right) C^h_{max}(s^*) - \left(\frac{m-3}{m-1}\right) + l_p$$

$$\le \left(2 - \frac{1}{m-1}\right) (\sum_{i=1}^{h} C^i_{max}(s^*) + l_p) - \left(\frac{m-3}{m-1}\right) \le \left(2 - \frac{1}{m-1}\right) C_{max}(s^*) - \left(\frac{m-3}{m-1}\right).$$

The tightness of (11) follows from the example presented in Figures 7, 8, and 9. Indeed, in this example (11) is tight for any k and m. □

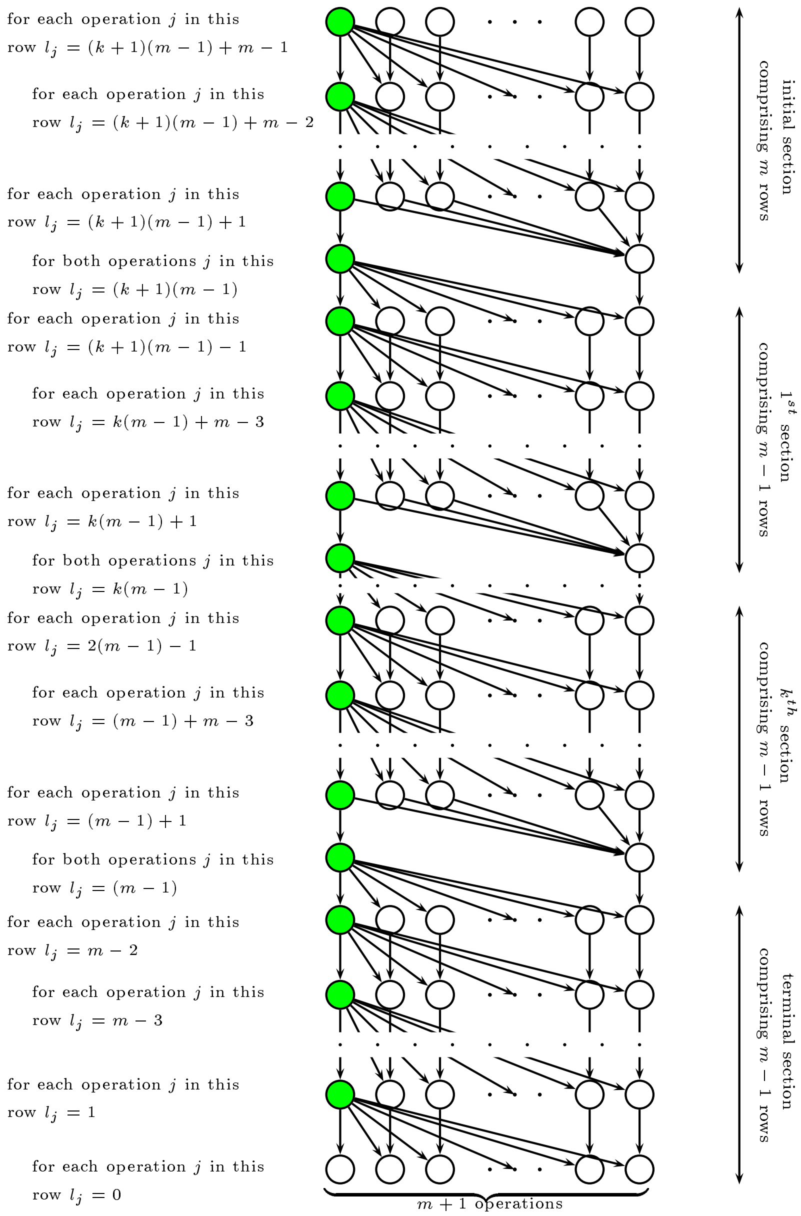

Figure 7: The partially ordered set of tasks considered in Theorems 2 and 3

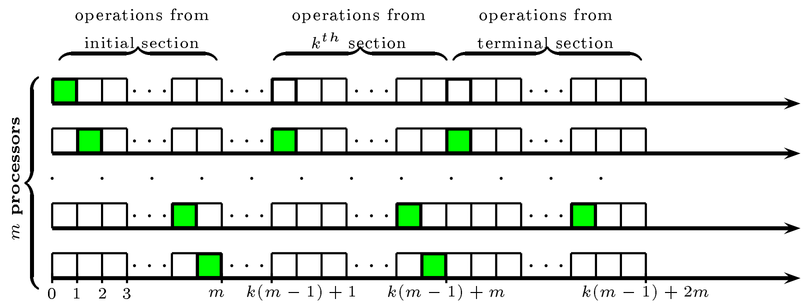

Figure 8: An optimal schedule for the makespan problem

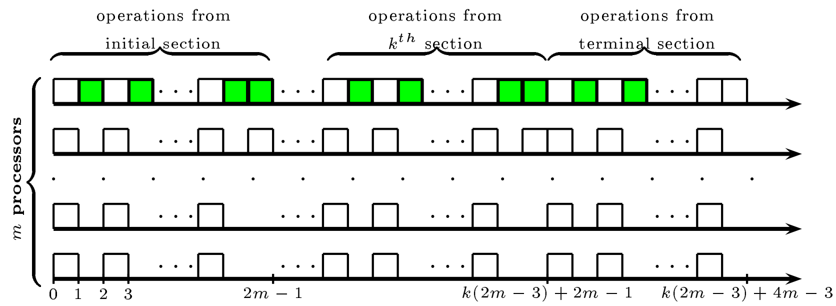

Figure 9: A non-optimal schedule constructed by Hu's algorithm

4 Conclusions

We have considered the makespan problem with a unit execution time task system, precedence constraints and parallel identical processors and tightened the well known performance guarantee (Chen 1975) for the classical Hu's algorithm. We have also introduced a tight performance guarantee which establishes the relationship between the deviation of the criterion value from its optimum and the parameters characterizing the problem. We have also considered the maximum lateness problem and presented a tight performance guarantee for the straightforward generalization of Hu's algorithm known as the Brucker-Garey-Johnson algorithm. A distinctive feature of all three presented bounds is their achievability for arbitrary large instances of the respective problems.

References

Braschi, B. and D. Trystram (1994) A new insight into the Coffman-Graham algorithm, *SIAM J. Comput.* **23**, 662-669.

Brucker, P., M.R. Garey and D.S. Johnson (1977) Scheduling equal-length tasks under tree-like precedence constraints to minimise maximum lateness, *Math. Oper. Res.* **2**, 275-284.

Chen, N.F. (1975), *An analysis of scheduling algorithms in multiprocessing computing sys-*

tems, Technical Report UIUCDCS-R-75-724, Department of Computer Science, University of Illinois at Urbana-Champaign.

Chen, N.F. and C.L. Liu (1975), On a class of scheduling algorithms for multiprocessors computing systems, in: T.Y. Feng(ed.), *Parallel Processing*, Lecture Notes in computer science, Vol. 24, Springer, Berlin, 1-16.

Coffman Jr, E.G. and R.L. Graham (1972) Optimal scheduling for two-processor systems, *Acta Inform.* **1**, 200-213.

Hu, T.C. (1961) Parallel sequencing and assembly line problems, *Operations Research* **9**, 841-848.

Lawler, E.L., J.K. Lenstra, A.H.G. Rinnooy Kan and D.B. Shmoys (1993) Sequencing and Scheduling: Algorithms and Complexity, in: *Logistics of Production and inventory*, eds. S. C. Graves, A. H. G. Rinnooy Kan and P. H. Zipkin, Elsevier Science Publishers B.V.

Lenstra, J.K. and A.H.G. Rinnooy Kan (1978) Complexity of scheduling under precedence constraints, *Oper. Res.* **26**, 22-35.

Parker, R.G. (1995) *Deterministic Scheduling Theory*, Chapman & Hall, New York.

Ullman, J.D. (1975) NP-Complete scheduling problems, *J. Comput. System Sci.* **10**, 384-393.

Zinder, Y. and D. Roper (1998), An iterative algorithm for scheduling unit-time operations with precedence constraints to minimise the maximum lateness, *Annals of Operations Research* **81**, 321-340.

GAURAV SINGH is a Ph.D. Student in the School of Mathematical Sciences, University of Technology, Sydney. He received his Master of Mathematics from Devi Ahilya Vishwa Vidhyalaya in 1997. He also received his Master of Science in Operations Research from the University of Technology, Sydney in 1998.

YAKOV ZINDER is a Senior Lecturer at the School of Mathematical Sciences in the University of Technology, Sydney. He holds a Ph.D. degree from the Academy of Sciences of Soviet Union and academic title Docent.