Dynamic Scheduling of Parallel Computations

Zhen Liu

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Dynamic Scheduling of Parallel Computations

Zhen LIU

N° 3048

Novembre 1996

THÈME 1

ISSN 0249-6399

INRIA
SOPHIA ANTIPOLIS

Dynamic Scheduling of Parallel Computations

Zhen LIU

Thème 1 — Réseaux et systèmes

Projet MISTRAL

Rapport de recherche n ° 3048 — Novembre 1996 — 17 pages

Abstract: Structures of parallel programs are usually modeled by task graphs in the scheduling literature. Such graphs are sometimes obtained while compiling the parallel programs. In many other cases, however, they can be determined only at run time. In this paper, we consider the scheduling of parallel computations whose task graphs are generated at run time. We analyze the case where the task graph has a random out-tree structure. When the number of offspring of a task has geometric distribution with a parameter which is decreasing and convex in the level, or the generation, then the breadth first policy stochastically minimizes the makespan. If, however, this parameter is increasing and concave, then the depth first policy stochastically minimizes the makespan.

Key-words: Dynamic Scheduling, Parallel Computation, Random Task Graph, Makespan, Schedule Length, Stochastic Optimization.

(Résumé : tsvp)

Correspondence: Zhen LIU, INRIA, Centre Sophia Antipolis, 2004 route des Lucioles, B.P. 93, 06902 Sophia-Antipolis, France. e-mail: Zhen.Liu@sophia.inria.fr

Unité de recherche INRIA Sophia Antipolis
2004 route des Lucioles, BP 93, 06902 SOPHIA ANTIPOLIS Cedex (France)
Téléphone : 04 93 65 77 77 – Télécopie : 04 93 65 77 65

Ordonnancement dynamique du calcul parallèle

Résumé : Les structures des programmes parallèles sont souvent modélisées par les graphes de tâches dans la littérature de l'ordonnancement. Tels graphes sont parfois obtenus lors de la compilation des programmes parallèles. Cependant, dans d'autres cas, ils ne peuvent être déterminés qu'au moment d'exécution. Dans cet article, nous considérons l'ordonnancement du calcul parallèle dont les graphes de tâches sont générés lors de leur exécution. Nous analysons le cas où la structure du graphe est une arborescence. Quand le nombre de descendants d'une tâche a une distribution géométrique avec un paramètre qui est décroissant et convexe en niveau, alors la politique "largeur d'abord" minimise stochastiquement la durée d'ordonnancement. Par contre, si ce paramètre est croissant et concave en niveau, alors la politique "profondeur d'abord" minimise stochastiquement la durée d'ordonnancement.

Mots-clé : ordonnancement dynamique, calcul parallèle, graphe de tâches aléatoire, durée d'ordonnancement, optimisation stochastique.

1 Introduction

Scheduling of parallel computations has been receiving increasing interest for more than two decades. Researchers from both the communities of parallel computing and of combinatorial optimization have obtained a number of results on the complexity of the problems and optimal solutions. The reader is referred to the books [1, 3] for the recent development in the field.

In this paper, we are interested in a scheduling problem in multiprocessor systems with identical parallel processors. The number of available processors can vary in time, so that we have a *variable profile*. Parallel programs running in the system are composed of sets of (sequential) tasks. The structure of a parallel program is represented by a *task graph* $G = (V, E)$, where vertices in V represent tasks, and arcs in E the precedence constraints between tasks. The scheduling problem is to assign tasks to parallel processors in such a way that the *makespan* (or *schedule length*) of a parallel program, i.e. the completion time of all the tasks of the program, is minimized. At any time, a processor can execute at most one task and a task can be run on at most one processor.

Much literature exists on scheduling of parallel programs with known deterministic structure. The interested reader is referred to [9] and [3] for surveys. In particular, the results on scheduling with variable profile and precedence constraints can be found in [10, 11] and the references therein.

However, in many applications such as branch-and-bound algorithms, structures of programs cannot be obtained in advance. Thus, "off-line" or static scheduling algorithms no longer apply. Instead, one has to use "on-line" or dynamic scheduling algorithms to deal with the dynamic structure of parallel programs, see [2] and [8] for discussions and references therein. A theoretical work was presented in [7] for the scheduling of dynamic binary trees on a ring of processors. Asymptotic optimality results were obtained when the number of tasks goes to infinity.

In this paper, we analyze the case where the dynamic structure of parallel programs is an out-tree. A parallel program starts with a single task. When a task finishes execution, it creates several other tasks, referred to as offspring. The numbers of offspring of tasks, which are possibly zero, are random variables. The running times of the tasks are constant with unit execution time (UET). The parallel program is completed if all its tasks have finished execution.

The computational model can be considered as an extension of that of [7]. However, we shall not take into account the interprocessor communication times when we analyze scheduling solutions. A similar computational model was considered in [13] in the framework of load balancing where jobs are represented by random out-trees and are to be assigned to different processors. Such a model was also used in the framework of performance evaluation of parallel computations see e.g. [5, 12].

Since the numbers of offspring of tasks are random variables, the resulting task graph of a parallel program is also random, even if the tasks are UET. Thus, for any schedule, the makespan of the task graph is a random variable too.

We present two optimal scheduling policies which stochastically minimize the makespan under specific statistical assumptions about the random graph. More precisely we show that when the number of offspring of a task has a geometric distribution with a parameter which is decreasing and convex in the level (or the generation), then the breadth first (BF) policy, also called highest-level first (HLF) or earliest-generation first (EGF), stochastically minimizes the makespan. When this parameter is increasing and concave, then the depth first (DF) policy, also called lowest-level first (LLF) or latest-generation first (LGF), stochastically minimizes the makespan.

In what follows, we first describe the problem in detail in section 2. We then state the results and provide the proofs in section 3. Concluding remarks are made in section 4.

2 Problem Description and Notation

The initial task of a parallel program is said to be of generation 0. Its offspring are of generation 1. In general, offspring of a task of generation i are of generation $i+1$, $i = 0, 1, \ldots$. The numbers of offspring of tasks are mutually independent random variables. For any fixed i, $i = 0, 1, \ldots$, the numbers of offspring of tasks of generation i are identically distributed, with generic random variable $N_i \in I\!N_+$, where $I\!N_+ = \{0, 1, 2, \ldots\}$. The random variable N_i has a geometric distribution with parameter q_i such that $P(N_i = n) = \bar{q}_i q_i^n$ for all $n \in I\!N_+$, where $P(A)$ denotes the probability of event A, and $\bar{q}_i = 1 - q_i$.

The multiprocessor system is composed of identical processors. The number of processors available, referred to as profile, for the execution of the parallel program varies in time. We assume that the profile varies only at integer time instants. These profile change epochs are arbitrarily random variables and are assumed to be independent of the random variables of the numbers of offspring of the tasks.

The running times of the tasks are constant with unit execution time (UET). Since the number of offspring of tasks are random variables, the resulting task graph G of a parallel program is also random, even if the profile is constant.

In this paper we consider dynamic, or on-line schedules, where all scheduling decisions are made with partial knowledge of the task graph. More precisely, the scheduler only knows the currently enabled tasks (those without unfinished predecessors) as well as their generations. The number of offspring of any task is unknown until it is finished.

No preemptive schedule is allowed so that once a processor starts execution of a task, it has to finish the execution before dealing with another task. Since the profile change occurs only at integer time instants, and the schedules are nonpreemptive, all schedules make decisions at integer time instants as well.

The following list schedules, referred to as breadth first (BF) and depth first (DF) denoted by β and δ, respectively, will be of particular interest. The BF policy can also be called highest-level first or earliest-generation first policies, and the DF policy called lowest-level first or latest-generation first policies.

At any time, schedule BF (resp. DF) arranges waiting tasks in a list in the increasing (resp. decreasing) order of their generations, with the task of the earliest (resp. latest) generation at the head of the list. As soon as a processor is available, the task at the head of the list is assigned to it. When new tasks are created, they are inserted in the list according to their generations. The execution of the program is finished when the list becomes empty.

Let $M_\sigma(G)$ be the makespan obtained by a nonpreemptive schedule σ on program G. Again, $M_\sigma(G)$ is a random variable.

We will prove that when the parameter q_i is decreasing and convex (resp. increasing and concave) in i, schedule BF (resp. DF) stochastically minimizes the makespan of a program for any arbitrary profile within the class of nonpreemptive schedules.

In order to prove the optimality of schedules of BF and DF, we need to use stochastic orders to compare random variables.

Let $I\!R_+ = [0, \infty)$. Random variable $X \in I\!R_+$ is said to be stochastically smaller than $Y \in I\!R_+$, denoted $X \leq_{\rm st} Y$, if for all $x \in I\!R_+$, $P(X > x) \leq P(Y > x)$. Stochastic dominance $\leq_{\rm st}$ implies dominance of moments: if $X \leq_{\rm st} Y$, then for all $n = 1, 2, \ldots$, $E[X^n] \leq E[Y^n]$.

The following result is due to Strassen [14].

Lemma 1 *Two random variables X and Y satisfy $X \leq_{\rm st} Y$ if and only if there exist two random variables $\hat{X}$ and $\hat{Y}$ defined on a common probability space such that $X =_{st} \hat{X}$, $Y =_{st} \hat{Y}$, and $\hat{X} \leq \hat{Y}$ almost surely (a.s.).*

3 Optimal Schedules for UET Tasks

We first consider the optimality of the BF schedule.

Theorem 2 *Assume that $q_i \leq 1/2$ for all $i \in I\!N_+$ and that q_i is decreasing and convex in i. Then schedule BF stochastically minimizes the makespan of the random task graph G within the class of nonpreemptive schedules, i.e., for any nonpreemptive schedule σ,*

$$M_\beta(G) \leq_{\rm st} M_\sigma(G).$$

Proof. Let the profile be arbitrarily fixed. Consider an arbitrary schedule σ for this profile. If σ makes BF decisions all the time, then clearly $M_\beta(G) =_{st} M_\sigma(G)$, so that the result trivially holds. Assume σ differs from β. Then, at some time, σ assigns a task, say task v, of generation j to an available processor, whereas there is a task, say task u, of generation i, $i < j$, waiting for execution.

Consider task graph G. Let $h \geq 0$ (resp. $k \geq 0$) be the number of offspring of u (resp. v) in G. Let $l = \min(h,k)$, $m = \max(h,k)$ and $n = m - l$. Denote by $u_1, \cdots, u_l$, and if $h > k$, $w_1, \cdots, w_n$, the offspring of u. Similarly, denote by $v_1, \cdots, v_l$, and if $k > h$, $w_1, \cdots, w_n$, the offspring of v. For $1 \leq r_1 \leq n$, let w_{r_1} have s_{1,r_1} offspring, denoted $w_{r_1,1}, \cdots, w_{r_1,s_{1,r_1}}$. For $1 \leq r_2 \leq s_{1,r_1}$, let w_{r_1,r_2} have s_{2,r_1,r_2} offspring, denoted $w_{r_1,r_2,1}, \cdots, w_{r_1,r_2,s_{2,r_1,r_2}}$. More generally, vertex $w_{r_1,\ldots,r_t}$ has $s_{t,r_1,\ldots,r_t}$ offspring, denoted $w_{r_1,\ldots,r_t,1}, \cdots, w_{r_1,\ldots,r_t,s_{t,r_1,\ldots,r_t}}$. Let there be T generations in total in the subtrees of $w_1, \cdots, w_n$. For $1 \leq t \leq T$, let $S_t = \sum_{r_1,\ldots,r_t} s_{t,r_1,\ldots,r_t}$, and $S_0 = n$.

Construct a new task graph G' as follows. G' is identical to G except for the subtrees whose roots are u and v. If $h \leq k$ then u (resp. v) has k (resp. h) offspring in G', denoted by $u_1, \cdots, u_l$ and $w_1, \cdots, w_n$ (resp. denoted by $v_1, \cdots, v_l$), see Figure 1. If $h > k$, see Figure 2, then, with probability p (see the definition below), u (resp. v) has k (resp. h) offspring in G', denoted by $u_1, \cdots, u_l$ (resp. denoted by $v_1, \cdots, v_l$ and $w_1, \cdots, w_n$), and with probability $1-p$, u (resp. v) has h (resp. k) offspring in G', denoted by $u_1, \cdots, u_l$ and $w_1, \cdots, w_n$ (resp. denoted by $v_1, \cdots, v_l$). The subtrees with roots $u_1, \cdots, u_l$, $v_1, \cdots, v_l$ and $w_1, \cdots, w_n$ are identical in G and G'.

The probability p is defined by

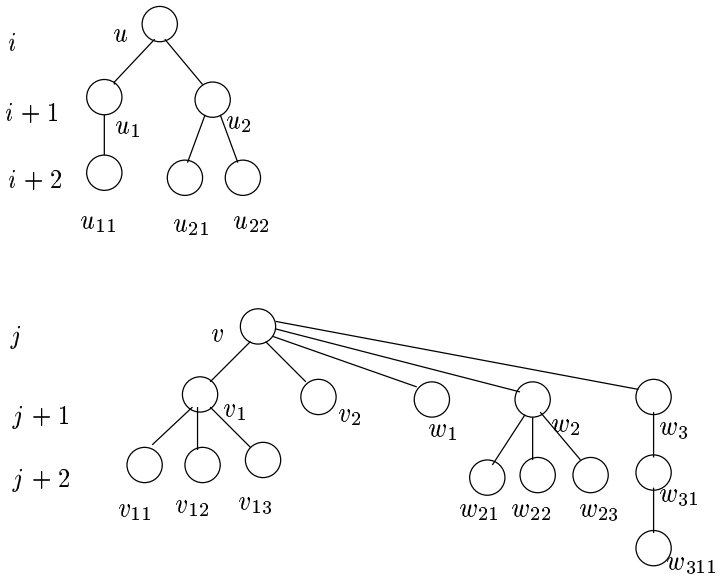
$$p = \prod_{t=0}^{T-1} \frac{(q_{j+t}\bar{q}_{j+t+1})^{S_t}}{(q_{i+t}\bar{q}_{i+t+1})^{S_t}} \tag{1}$$

For the example in Figures 1 and 2,

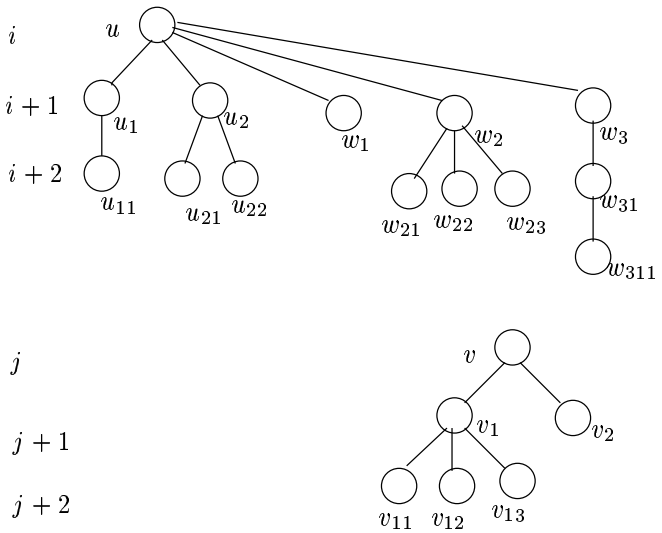
$$p = \frac{q_j^2 \bar{q}_{j+1}^2 q_{j+1}^3 \bar{q}_{j+2}^3 q_{j+2} \bar{q}_{j+3}}{q_i^2 \bar{q}_{i+1}^2 q_{i+1}^3 \bar{q}_{i+2}^3 q_{i+2} \bar{q}_{i+3}}. \tag{2}$$

We will show that the graph G' has the same distribution as G. Let us start with proving that p defined by (1) is smaller than 1. Under the assumptions of the theorem, it is easy to see that for any $t \geq 0$ and $i < j$,

$$q_{i+t} - q_{j+t} \geq 0, \qquad \text{decreasingness of } q$$



Graph G



Graph G'

Figure 1: Construction of graph G': case $h < k$.

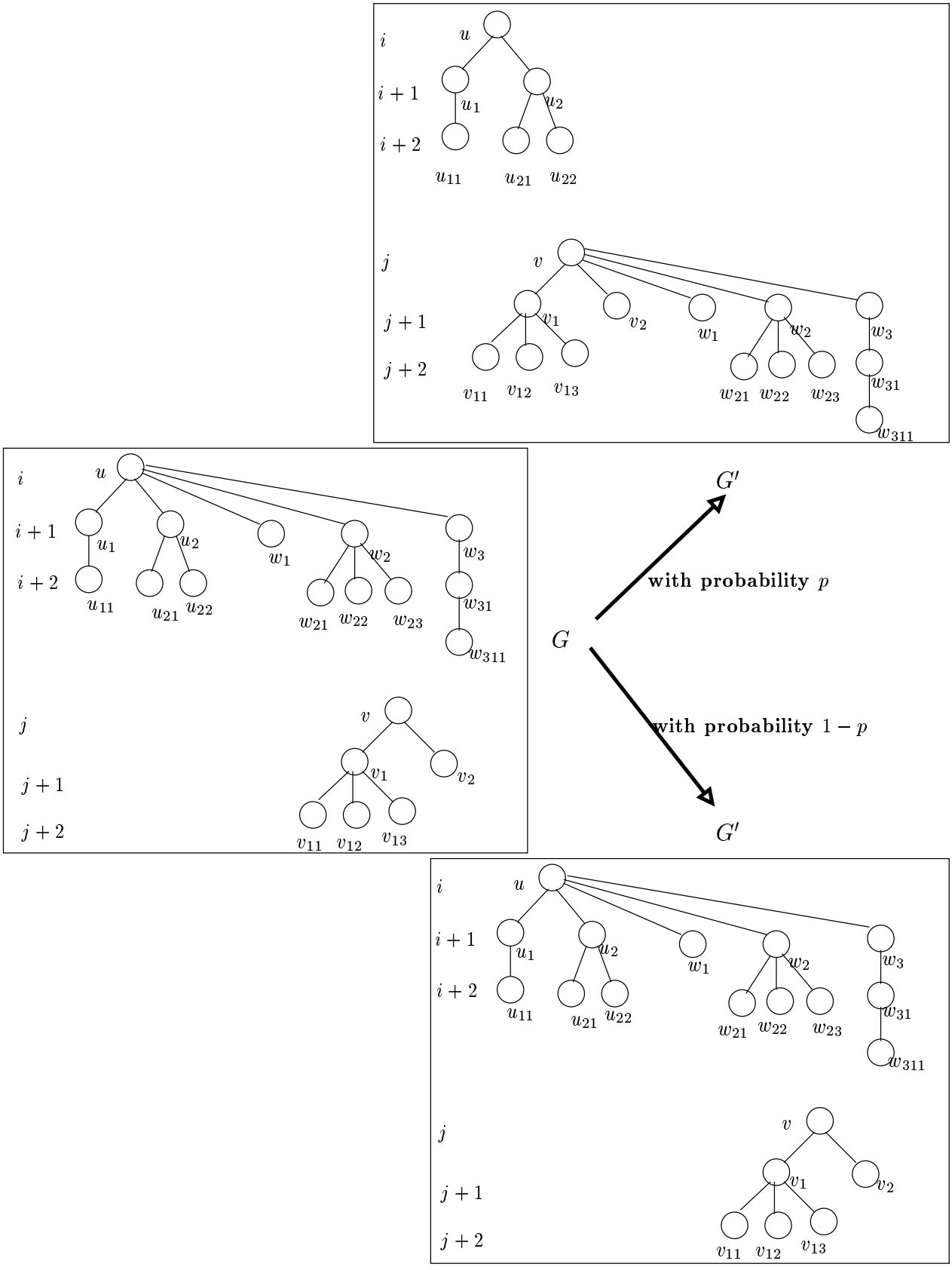


Figure 2: Construction of graph G': case $h > k$.

$$q_{i+t+1} - q_{j+t+1} \leq q_{i+t} - q_{j+t}, \qquad \text{convexity of } q$$

Thus,

$$\begin{aligned}
& q_{i+t}\bar{q}_{i+t+1} - q_{j+t}\bar{q}_{j+t+1} \\
&= q_{i+t} - q_{i+t}q_{i+t+1} - q_{j+t} + q_{j+t}q_{j+t+1} \\
&= q_{i+t} - q_{j+t} - q_{i+t}q_{i+t+1} + q_{j+t}q_{i+t+1} - q_{j+t}q_{i+t+1} + q_{j+t}q_{j+t+1} \\
&\geq q_{i+t} - q_{j+t} - q_{i+t+1}(q_{i+t} - q_{j+t}) - q_{j+t}(q_{i+t} - q_{j+t}) \\
&= (q_{i+t} - q_{j+t})(1 - q_{i+t+1} - q_{j+t}) \\
&\geq 0,
\end{aligned}$$

where we used the fact that q's are bounded by $1/2$ in the last inequality. Therefore, the quantity p defined by (1) is upper bounded by 1.

In order to show that G' has the same distribution as G, we only need to show that the subtrees with roots u and v have the same distribution in G and G'.

Let $N(x)$ be the number of offspring of vertex x. Let A be an event of the forest rooted by vertices $u_1, \ldots, u_l$, and B an event of the forest rooted by vertices $v_1, \ldots, v_l$. For example, in Figures 1 and 2, A is the event that $\{N(u_1) = 1, N(u_2) = 2, N(u_{11}) = N(u_{21}) = N(u_{22}) = 0\}$, and B the event that $\{N(v_1) = 3, N(v_2) = N(v_{11}) = N(v_{12}) = N(v_{13}) = 0\}$.

Let C be the event of the forest rooted by vertices $w_1, \ldots, w_n$, where there are in total T generations, and for $1 \leq r_1 \leq n$, w_{r_1} has s_{1,r_1} offspring, and for $1 \leq r_2 \leq s_{1,r_1}$, w_{r_1,r_2} has s_{2,r_1,r_2} offspring, etc. More generally, for $1 \leq t \leq T$, vertex $w_{r_1,\ldots,r_t}$ has $s_{t,r_1,\ldots,r_t}$ offspring, denoted $w_{r_1,\ldots,r_t,1}, \cdots, w_{r_1,\ldots,r_t,s_{t,r_1,\ldots,r_t}}$. Let $S_t = \sum_{r_1,\ldots,r_t} s_{t,r_1,\ldots,r_t}$, and $S_0 = n$.

For the example in Figures 1 and 2, C is the event that $\{T = 3, N(w_1) = 0, N(w_2) = 3, N(w_3) = 1, N(w_{21}) = N(w_{22}) = N(w_{23}) = 0, N(w_{31}) = 1, N(w_{311}) = 0\}$.

We first compute the probability distribution of C for graph G, denoted as $P(C)$. Under the assumption that the numbers of successors of tasks are mutually independent random variables, $P(C)$ has a product-form expression. For example, in Figure 1,

$$\begin{aligned}
& P(C) \\
&= P(N(w_1) = 0, N(w_2) = 3, N(w_3) = 1, \\
&\quad N(w_{21}) = N(w_{22}) = N(w_{23}) = 0, N(w_{31}) = 1, N(w_{311}) = 0) \\
&= P(N_{j+1} = 0)P(N_{j+1} = 3)P(N_{j+1} = 1) \\
&\quad P(N_{j+2} = 0)P(N_{j+2} = 0)P(N_{j+2} = 0)P(N_{j+2} = 1)P(N_{j+3} = 0)
\end{aligned}$$

$$
\begin{aligned}
&= \bar{q}_{j+1}(\bar{q}_{j+1}q_{j+1}^3)(\bar{q}_{j+1}q_{j+1})\bar{q}_{j+2}^3(\bar{q}_{j+2}q_{j+2})\bar{q}_{j+3} \\
&= \bar{q}_{j+1}^3 q_{j+1}^4 \bar{q}_{j+2}^4 q_{j+2}\bar{q}_{j+3},
\end{aligned}
$$

and in Figure 2,

$$
\begin{aligned}
&P(C) \\
&= P(N(w_1)=0, N(w_2)=3, N(w_3)=1, \\
&\quad N(w_{21})=N(w_{22})=N(w_{23})=0, N(w_{31})=1, N(w_{311})=0) \\
&= P(N_{i+1}=0)P(N_{i+1}=3)P(N_{i+1}=1) \\
&\quad P(N_{i+2}=0)P(N_{i+2}=0)P(N_{i+2}=0)P(N_{i+2}=1)P(N_{i+3}=0) \\
&= \bar{q}_{i+1}(\bar{q}_{i+1}q_{i+1}^3)(\bar{q}_{i+1}q_{i+1})\bar{q}_{i+2}^3(\bar{q}_{i+2}q_{i+2})\bar{q}_{i+3} \\
&= \bar{q}_{i+1}^3 q_{i+1}^4 \bar{q}_{i+2}^4 q_{i+2}\bar{q}_{i+3}.
\end{aligned}
$$

In general, if w-tasks are successors of v, then

$$
P(C) = \left(\prod_{t=1}^{T-1}\left(\bar{q}_{j+t}^{S_{t-1}} q_{j+t}^{S_t}\right)\right)\cdot \bar{q}_{j+T}^{S_{T-1}} = q_j^{-n}\prod_{t=0}^{T-1}\left(q_{j+t}\bar{q}_{j+t+1}\right)^{S_t} \tag{3}
$$

and if w's are successors of u, then

$$
P(C) = \left(\prod_{t=1}^{T-1}\left(\bar{q}_{i+t}^{S_{t-1}} q_{i+t}^{S_t}\right)\right)\cdot \bar{q}_{i+T}^{S_{T-1}} = q_i^{-n}\prod_{t=0}^{T-1}\left(q_{i+t}\bar{q}_{i+t+1}\right)^{S_t} \tag{4}
$$

Denote by P' the probability distribution of graph G'.

Then, in case that u has a smaller number of offspring in G' (cf. Figure 2), we have

$$
\begin{aligned}
&P'(N(u)=l, N(v)=m, A, B, C) \\
&= pP(N(u)=m, N(v)=l, A, B, C) \\
&= pP(A)P(B)(\bar{q}_i q_i^m)(\bar{q}_j q_j^l)q_i^{-n}\prod_{t=0}^{T-1}\left(q_{i+t}\bar{q}_{i+t+1}\right)^{S_t} \qquad \text{from (4)} \\
&= P(A)P(B)(\bar{q}_i q_i^m)(\bar{q}_j q_j^l)q_i^{-n}\prod_{t=0}^{T-1}\left(q_{j+t}\bar{q}_{j+t+1}\right)^{S_t} \qquad \text{from (1)} \\
&= P(A)P(B)(\bar{q}_i q_i^l)(\bar{q}_j q_j^m)q_j^{-n}\prod_{t=0}^{T-1}\left(q_{j+t}\bar{q}_{j+t+1}\right)^{S_t} \\
&= P(N(u)=l, N(v)=m, A, B, C)\ . \qquad \text{from (3)}
\end{aligned}
$$

Thus, the probability distributions P' and P are the same.

In the case that u has larger number of offspring in G', we have

$$\begin{aligned}
&P'(N(u)=m,N(v)=l,A,B,C)\\
&= P(N(u)=l,N(v)=m,A,B,C)\\
&\quad +(1-p)P(N(u)=m,N(v)=l,A,B,C)\\
&= P(A)P(B)(\bar{q}_i q_i^l)(\bar{q}_j q_j^m)q_j^{-n}\prod_{t=0}^{T-1}(q_{j+t}\bar{q}_{j+t+1})^{S_t}\\
&\quad +P(A)P(B)(1-p)(\bar{q}_i q_i^m)(\bar{q}_j q_j^l)q_i^{-n}\prod_{t=0}^{T-1}(q_{i+t}\bar{q}_{i+t+1})^{S_t} \qquad \text{from (3)and (4)}\\
&= P(A)P(B)(\bar{q}_i q_i^m)(\bar{q}_j q_j^l)q_i^{-n}\left\{\prod_{t=0}^{T-1}(q_{j+t}\bar{q}_{j+t+1})^{S_t}+(1-p)\prod_{t=0}^{T-1}(q_{i+t}\bar{q}_{i+t+1})^{S_t}\right\}\\
&= P(A)P(B)(\bar{q}_i q_i^m)(\bar{q}_j q_j^l)q_i^{-n}\prod_{t=0}^{T-1}(q_{i+t}\bar{q}_{i+t+1})^{S_t} \qquad \text{from (1)}\\
&= P(N(u)=m,N(v)=l,A,B,C)\ . \qquad \text{from (4)}
\end{aligned}$$

Thus, again, the probability distributions P' and P are the same.

We now define a new schedule ρ on graph G' as follows. Schedule ρ is identical to σ except for the two subtrees with roots u and v. Schedule ρ interchanges the assignment decisions of σ for u and v, i.e., when σ assigns v (resp. u) to an available processor, ρ assigns u (resp. v) to the processor. The assignment decisions for the vertices of subtrees with roots $w_1,\cdots,w_n$ are kept the same under ρ as under σ. For each pair of vertices (u_r,v_r), $1\le r\le l$, if u_r is assigned to a processor no later than v_r under σ, then ρ follows the same assignments. If, however, u_r is assigned to a processor strictly later than v_r under σ, then ρ interchanges the assignment decisions of σ for these two vertices, and we construct new subtrees with roots u_r and v_r, in a similar way as we do for the pair of vertices (u,v).

One can check that this new schedule ρ is a feasible schedule in the sense that a vertex is assigned to a processor only after its predecessor is finished. Moreover, it is easily seen that

$$M_\rho(G')=M_\sigma(G).$$

Note that this new schedule is possibly an idling schedule due to the fact the w-tasks are enabled earlier. Thus, we can have a nonidling schedule ρ' by removing some idle periods in ρ so that the makespan is decreased. Thus,

$$M_{\rho'}(G')\le M_\sigma(G).$$

As G' and G have the same probabilistic distribution, we conclude that

$$M_{\rho'}(G)\le_{\rm st} M_\sigma(G).$$

One can easily see that ρ' has less non-BF decisions than σ. An inductive argument yields

$$M_\beta(G) \leq_{\rm st} M_\sigma(G).$$

■

In a similar way, we can show the optimality of the DF schedule.

Theorem 3 *Assume that $q_i \leq 1/2$ for all $i \in I\!N_+$ and that q_i is increasing and concave in i. Then schedule DF stochastically minimizes the makespan of the random task graph G within the class of nonpreemptive schedules, i.e., for any nonpreemptive schedule σ,*

$$M_\delta(G) \leq_{\rm st} M_\sigma(G).$$

Proof. The proof is similar to the previous one. We shall only sketch the arguments.

Again, let the profile be arbitrarily fixed. Consider an arbitrary schedule σ for this profile. Assume σ differs from δ. Then, at some time, σ assigns a task, say task u, of generation i to an available processor, whereas there is a task, say task v, of generation j, $j > i$, waiting for execution.

Consider task graph G. We use the same notation as in the proof of Theorem 2: $h \geq 0$ (resp. $k \geq 0$) is the number of offspring of u (resp. v) in G, and $l = \min(h,k)$, $m = \max(h,k)$ and $n = m - l$. The offspring of u are $u_1, \cdots, u_l$, and if $h > k$, $w_1, \cdots, w_n$, and those of v are $v_1, \cdots, v_l$, and if $k > h$, $w_1, \cdots, w_n$. There are T generations in total in the subtrees of $w_1, \cdots, w_n$, with S_t being the number of offspring of tasks of generation t, $0 \leq t \leq T$, and $S_0 = n$.

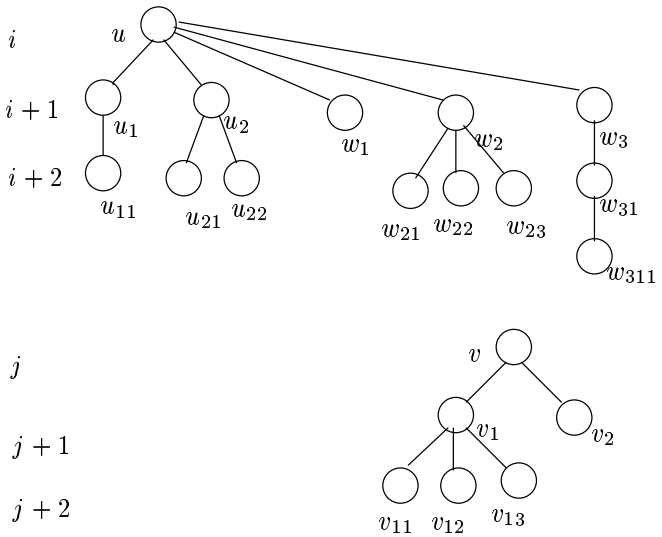
Construct a new task graph G' as follows. G' is identical to G except for the subtrees whose roots are u and v. If w-tasks are offspring of u in G, then these w-tasks become the offspring of v in G', see Figure 3. If w-tasks are offspring of v in G, then, with probability p (to be defined below in (5)), these w-tasks become the offspring of u in G', and with probability $1-p$ these w-tasks remain the offspring of v in G', see Figure 4. The subtrees with roots $u_1, \cdots, u_l$, $v_1, \cdots, v_l$ and $w_1, \cdots, w_n$ are identical in G and G' for all these cases.

The probability p is defined by

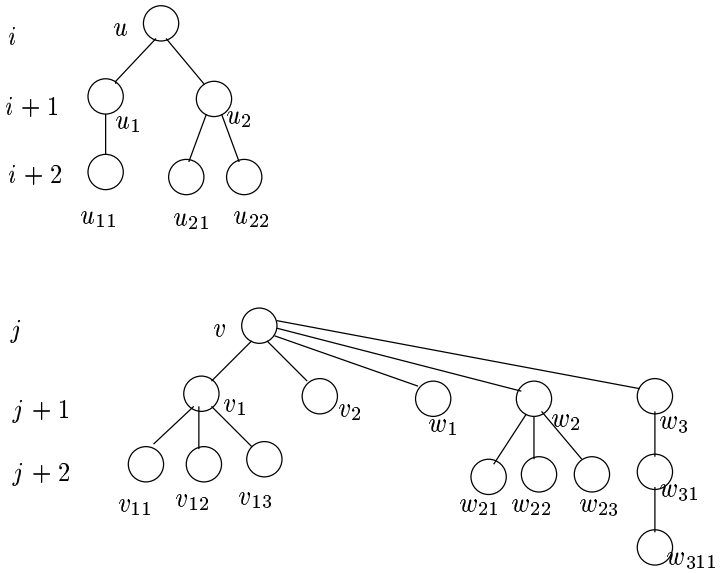
$$p = \prod_{t=0}^{T-1} \frac{(q_{i+t}\bar{q}_{i+t+1})^{S_t}}{(q_{j+t}\bar{q}_{j+t+1})^{S_t}} \tag{5}$$

The quantity p defined in this way is again upper bounded by 1 under the assumptions of the theorem. Indeed, for any $t \geq 0$ and $i < j$,

$$q_{j+t} - q_{i+t} \geq 0, \qquad \text{increasingness of } q$$



Graph G



Graph G'

Figure 3: Construction of graph G': case $h < k$.

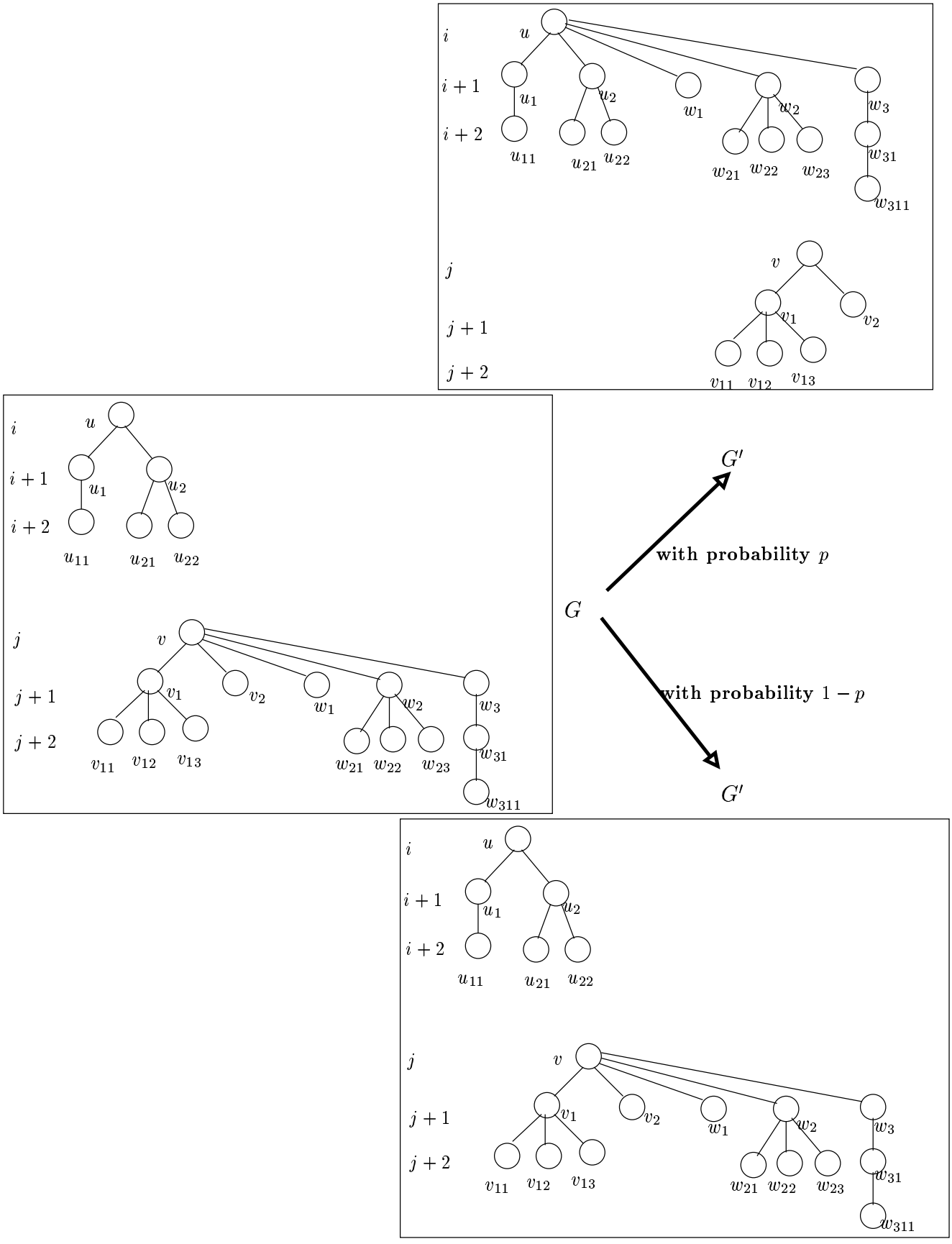


Figure 4: Construction of graph G': case $h > k$.

$$q_{j+t+1} - q_{i+t+1} \leq q_{j+t} - q_{i+t}, \qquad \text{concavity of } q$$

Thus,

$$\begin{aligned}
& q_{j+t}\bar{q}_{j+t+1} - q_{i+t}\bar{q}_{i+t+1} \\
&= q_{j+t} - q_{j+t}q_{j+t+1} - q_{i+t} + q_{i+t}q_{i+t+1} \\
&= q_{j+t} - q_{i+t} - q_{j+t}q_{j+t+1} + q_{i+t}q_{j+t+1} - q_{i+t}q_{j+t+1} + q_{i+t}q_{i+t+1} \\
&\geq q_{j+t} - q_{i+t} - q_{j+t+1}(q_{j+t} - q_{i+t}) - q_{i+t}(q_{j+t} - q_{i+t}) \\
&= (q_{j+t} - q_{i+t})(1 - q_{j+t+1} - q_{i+t}) \\
&\geq 0.
\end{aligned}$$

By mimicking the previous proof one can show that the graph G' has the same distribution as G. The remaining proof is analogous to that of Theorem 2 and is omitted. ∎

4 Concluding Remarks

In this paper we have shown that for parallel computations represented as random out trees, the breadth first (resp. depth first) policy is optimal within the class of dynamic nonpreemptive schedules for the stochastic minimization of the makespan when the numbers of offspring of tasks are geometrically distributed with the parameters being decreasing and convex (resp. increasing and concave).

Moreover, in contrast to the static scheduling, our result holds for any arbitrary variable profile. Indeed, it is well-known that the minimization of the makespan of UET tasks with tree structure is *NP*-hard under general profile. The interested reader is referred to Garey et al. [6] who actually proved the NP-completeness of $P(t)\ decreasing \mid p_i = 1, intree \mid C_{\max}$ (so is $P(t)\ increasing \mid p_i = 1, outtree \mid C_{\max}$), where the decreasing profile refers to the case where the number of available processors is decreasing in time. However, as we assumed the geometric distributions of the numbers of offspring, the resulting outtree has a particular structure which is related to the class of uniform-outforests introduced in [4].

When preemption is allowed, our result can be extended to the case where task running times are random variables with an exponential distribution. The static-graph version of the problem has been investigated in [4].

Although the work presented in the paper can be considered as one of the first theoretical results on the makespan minimization of parallel computations with dynamic precedence constraints, the

computational model used here is quite restrictive. We need to extend the model in order to deal with more realistic problems.

Another future research direction is the on-line scheduling of random task graphs with interprocessor communications. Interesting asymptotic results were obtained in [7] for binary trees when processors connected by a ring. For the same architecture but more general structure of out-trees, heuristics were evaluated empirically in [8].

Acknowledgements: The author is grateful to Professors Denis Trystram of INPG and Lixin Gao of Smith College for their useful comments on the work.

References

[1] J. Blazewicz, K. Ecker, G. Schmidt, J. Weglarz, *Scheduling in Computer and Manufacturing Systems*, Springer-Verlag, 1993.

[2] J. Briat, T. Gautier, and J.-L. Roch, "On-line scheduling", *Proc. of ESPPE'96, Parallel Programming Environments for High Performance Computing*, pp. 95–108, April 1996.

[3] P. Chretienne, E. G. Coffman, J. K. Lenstra, Z. Liu, (Eds.) *Scheduling Theory and Its Applications*, J. Wiley, 1995.

[4] E. G. Coffman, Z. Liu, "On the Optimal Stochastic Scheduling of Out-Forests." *Operations Research*, Vol. 40, Supp. No. 1, pp. S67–S75, Jan. 1992.

[5] G. Fayolle, P. J. B. King, I. Mitrani, "On the Execution of Programs by Many Processors", *Proc. of Performance 83*, pp. 217-228, 1983.

[6] M. R. Garey, D. S. Johnson, R. E. Tarjan et M. Yannakakis, "Scheduling opposite forests", *SIAM J. Alg. Disc. Meth.*, Vol. 4, pp 72–93, 1983.

[7] L. Gao, A. L. Rosenberg, "Toward efficient scheduling of evolving computations on rings of processors", to appear in *J. Parallel and Distributed Computing.*

[8] D. E. Gregory, L. Gao, A. L. Rosenberg and P.R. Cohen, "An empirical study of dynamic scheduling on rings of processors", *Proc. of the 8th IEEE Symp. on Parallel and Distributed Processing*, pp. 470-473, 1996

[9] E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan and D. B. Shmoys, "Sequencing and scheduling: algorithms and complexity", Report BS-R8909, CWI, Amsterdam, Holland, 1989.

[10] Z. Liu, E. Sanlaville, "Preemptive Scheduling with Variable Profile, Precedence Constraints and Due Dates", *Discrete Applied Mathematics*, Vol. 58, pp. 253-280, April 1995.

[11] Z. Liu, E. Sanlaville, "Profile Scheduling by List Algorithms." In *Scheduling Theory and Its Applications*, P. Chretienne et al. (Eds.), J. Wiley, 1995, pp. 95-114.

[12] P. Mussi, P. Nain, "Evaluation of Parallel Execution of Program Tree Structures", *ACM Sigmetrics, Performance Evaluation Review*, Special Issue Vol.12 No.3, pp.78-87, August 1984.

[13] D. M. Nicol, R. Simha, D. Towsley, "Static Assignment of Stochastic Tasks Using Majorization", *IEEE Trans. on Computers*, Vol. 45, pp. 730–740, 1996.

[14] V. Strassen, "The Existence of Probability Measures with Given Marginals", *Ann. Math. Stat.*, Vol. 336, pp. 423-439, 1965.

Unité de recherche INRIA Lorraine, Technopôle de Nancy-Brabois, Campus scientifique,
615 rue du Jardin Botanique, BP 101, 54600 VILLERS LÈS NANCY
Unité de recherche INRIA Rennes, Irisa, Campus universitaire de Beaulieu, 35042 RENNES Cedex
Unité de recherche INRIA Rhône-Alpes, 655, avenue de l'Europe, 38330 MONTBONNOT ST MARTIN
Unité de recherche INRIA Rocquencourt, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex
Unité de recherche INRIA Sophia Antipolis, 2004 route des Lucioles, BP 93, 06902 SOPHIA ANTIPOLIS Cedex

Éditeur
INRIA, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex (France)
ISSN 0249-6399