An Efficient Algorithm for a Task Allocation Problem

A. BILLIONNET

CEDRIC, Institut d'Informatique d'Entreprise, Evry, France

M. C. COSTA

CEDRIC, Conservatoire National des Arts et Métiers, Paris, France

AND

A. SUTTER

France Telecom, Issy-Les-Moulineaux, France

Abstract. This paper presents an efficient algorithm to solve one of the task allocation problems. Task assignment in an heterogeneous multiple processors system is investigated. The cost function is formulated in order to measure the intertask communication and processing costs in an uncapacited network. A formulation of the problem in terms of the minimization of a submodular quadratic pseudo-Boolean function with assignment constraints is then presented. The use of a branch-and-bound algorithm using a Lagrangean relaxation of these constraints is proposed. The lower bound is the value of an approximate solution to the Lagrangean dual problem. A zero-duality gap, that is, a saddle point, is characterized by checking the consistency of a pseudo-Boolean equation. A solution is found for large-scale problems (e.g., 20 processors, 50 tasks, and 200 task communications or 10 processors, 100 tasks, and 300 task communications). Excellent experimental results were obtained which are due to the weak frequency of a duality gap and the efficient characterization of the zero-gap (for practical purposes, this is achieved in linear time). Moreover, from the saddle point, it is possible to derive the optimal task assignment.

Categories and Subject Descriptors: C.4 [**Computer Systems Organization**]: Performance of Systems—*modeling techniques*, D 4.1 [**Operating Systems**]· Process Management—*multiprocessing*; D 4 7 [**Operating Systems**]: Organization and Design—*distributed systems*; G.2.1 [**Discrete Mathematics**]: Combinatorics—*combinatorial algorithms*

General Terms: Algorithms, Design, Measurement, Performance

Additional Key Words and Phrases Branch-and-bound algorithm, interprocessor communication, Lagrangean relaxation, quadratic 0–1 optimization, task allocation

Authors' addresses: A. Billionnet, CEDRIC, Institut d'Informatique d'Entreprise, 18 Allée Jean Rostand, 91025 Evry Cedex, France; M. C Costa, CEDRIC, Conservatoire National des Arts et Métiers, 292 rue Saint Martin, 75141 Paris Cedex 03, France; A. Sutter, France Telecom, Centre National d'Etudes des Télécommunications, CNET/PAA/ATR, 38–40 rue du Général Leclerc, 92131 Issy-Les-Moulineaux, Cedex, France.

1. *Introduction*

An important problem that arises in distributed computer systems is the so-called task allocation problem. Many heuristic approaches that provide suboptimal solutions have been attempted in a number of studies (3–7, 9–13, 15, 16]. However, for practical problems, it is difficult to evaluate how accurate these solutions are, because one does not know efficient algorithms that generate an exact solution. Indeed, up to now, all the exact allocation algorithms were limited to very-small-sized problems.

We describe in this paper an exact algorithm applicable to large-sized problems. It has been developed to solve this task allocation problem in a distributed computer system that meets the following specifications:

a) the processors of the system are heterogeneous. A single program module, if executed on different processors, will therefore require different amounts of running time;
b) identical communication links are used by the processors for message transmission. This means identical messages, even if transmitted through different communication links, will have identical transmission times;
c) the capacities of processors and links are assumed to be unlimited.

In the case of a two-processors system, it has been shown that the optimal assignment may be found very efficiently [17] by a polynomial-time algorithm. However, for an arbitrary number of processors, the problem is known to be NP-complete [10].

In Section 2 of this paper, we present a precise definition of the task allocation problem and we formulate it as the minimization of a quadratic pseudo-Boolean function with linear constraints. In Section 3, we study the Lagrangean dual problem and we show how it can be written as a continuous linear program. In Section 4, we propose an algorithm for computing an approximate dual solution and in Section 5, a procedure for testing the nonexistence of a duality gap (i.e., is the lower bound, computed in Section 4, actually equal to the optimum?). In Section 6, we describe the task assignment algorithm. It is of the branch-and-bound type that makes use of the previous theoretical results. Experimental results for a number of pseudo-randomly generated test cases are tabulated. It appears that our algorithm is applicable to large-size problems (e.g., 20 processors, 50 tasks, and 200 pairs of communicating tasks or 10 processors, 100 tasks, and 300 communicating tasks).

2. *Problem Definition and Modeling*

Let $\mathbf{P} = \{P_1, P_2, \ldots, P_m\}$ be the set of the nonidentical processors of the distributed system. A distributed process is defined as the set of tasks $\mathbf{T} = \{T_1, T_2, \ldots, T_n\}$ to be run on the distributed system. Some of these tasks have to communicate. These intertask communications are represented by the graph $\mathbf{G} = (\{1, \ldots, n\}, U)$ where $[i, j] \in U$ if and only if tasks T_i and T_j communicate with each other.

Let

$c_{tt'}$ $([t, t'] \in U)$ be the communication cost between two tasks $T_t \in \mathbf{T}$ and $T_{t'} \in \mathbf{T}$ if they are assigned to different processors. We assume that the communication cost between two tasks executed by the same processor is negligible;

q_{tp} $(t \in \{1, \ldots, n\}, p \in \{1, \ldots, m\})$ be the execution cost of task T_t when it is assigned to processor P_p;

x_{tp} $(t \in \{1, \dots, n\},\ p \in \{1, \dots, m\})$ be the decision Boolean variable that is equal to 1 if task T_t is assigned to processor P_p and 0, otherwise; $1 - x_{tp}$ is denoted by $\bar{x}_{tp}$.

We can now formulate the 0-1 programming problem to be solved:

$$\left.\begin{array}{lll} \text{Min} & \displaystyle\sum_{t=1}^{n} \sum_{p=1}^{m} q_{tp} x_{tp} + \sum_{\substack{t,t' \text{ such that} \\ [t,t'] \in U, t<t'}} \sum_{p=1}^{m} c_{tt'} x_{tp} \bar{x}_{t'p} & \\ \text{subject to} & \displaystyle\sum_{p=1}^{m} x_{tp} = 1 \qquad (t = 1, \dots, n) & \\ & x_{tp} \in \{0, 1\} \qquad (t = 1, \dots, n;\ p = 1, \dots, m). & \end{array}\right\} \quad (2.1)$$

The first summation term of the objective function represents the global processing cost; the second the intertask communication costs. The constraints are assignment constraints; that is, each task must be assigned to one and only one processor. Our purpose is to make the best use of resources in this distributed system, that is, for a given distributed process to minimize execution and communication costs. We do not therefore take into account precedence relationships among tasks. In the distributed system considered, the transmission cost of a given message between two processors does not depend upon these processors because either the network is a completely meshed network or it is a local network, and the differences between the costs are negligible. The cost of communication between two modules assigned to different processors is not therefore a function of the processors to which the modules are assigned.

Let us show that Program (2.1) can be stated as the minimization with constraints of a pseudo-Boolean function whose coefficients of the nonlinear terms are negative. Replacing $\bar{x}_{t'p}$ by $1 - x_{t'p}$ and denoting

$$l_{tp} = q_{tp} + \sum_{\substack{t' \text{ such that} \\ [t,t'] \in U, t<t'}} c_{tt'}$$

Program (2.1) can be written as follows:

$$\left.\begin{array}{lll} \text{Min} & \displaystyle\sum_{t=1}^{n} \sum_{p=1}^{m} l_{tp} x_{tp} - \sum_{\substack{t,t' \text{ such that} \\ [t,t'] \in U, t<t'}} \sum_{p=1}^{m} c_{tt'} x_{tp} x_{t'p} & \\ \text{subject to} & \displaystyle\sum_{p=1}^{m} x_{tp} = 1 \qquad (t = 1, \dots, n) & (2.2.1) \\ & x_{tp} \in \{0, 1\} \qquad (t = 1, \dots, n;\ p = 1, \dots, m). & \end{array}\right\} \quad (2.2)$$

It is known [14] that the minimum of the objective function in (2.2), without constraints, can be efficiently computed by using a maximum-flow algorithm on a bipartite graph. It seems natural, therefore, to consider the classical Lagrangean dual problem of (2.2) obtained by dualizing the assignment constraints. That will be considered in the following section.

Example. Let us consider a task allocation problem with three processors and four tasks. Figure 1 gives the execution costs and the intertask communication graph including the communication costs. The primal problem is

$$\begin{aligned}\text{Min}\quad & 11x_{11} + 12x_{12} + 16x_{13} + 17x_{21} + 8x_{22} + 9x_{23} + 10x_{31} \\ & +3x_{32} + x_{33} + 3x_{41} + 4x_{42} + x_{43} - 10x_{11}x_{41} - 10x_{12}x_{42} \\ & -10x_{13}x_{43} - 2x_{21}x_{41} - 2x_{22}x_{42} - 2x_{23}x_{43} - 5x_{21}x_{31} \\ & -5x_{22}x_{32} - 5x_{23}x_{33} \\ \text{subject to}\quad & \sum_{p=1}^{3} x_{tp} = 1 \qquad (t = 1,2,3,4) \\ & x_{tp} \in \{0,1\} \qquad (t = 1,2,3,4;\ p = 1,2,3)\end{aligned}$$

3. *The Lagrangean Dual Problem*

The solution of the Lagrangean dual problem gives a lower bound for the primal problem (2.2), which will be used in a branch-and-bound algorithm. Let us associate the Lagrange multipliers $\pi_t \in \mathbb{R}$ $(t = 1,\dots,n)$ with the constraints (2.2.1). The Lagrangean function is:

$$L(x,\pi) = -\sum_{t=1}^{n} \pi_t + \sum_{t=1}^{n}\sum_{p=1}^{m} (l_{tp} + \pi_t)x_{tp} - \sum_{\substack{t,t' \text{ such that,}\\ [t,t']\in U, t<t'}} \sum_{p=1}^{m} c_{tt'}\, x_{tp}x_{t'p}$$

with $\pi \in \mathbb{R}^n$ and $x \in \{0,1\}^{n\,m}$.

Then, the Lagrangean dual problem is:

$$\text{LD}^* = \underset{\pi\in\mathbb{R}^n}{\text{Max}} \left[\underset{x\in\{0,1\}^{n\,m}}{\text{Min}} L(x,\pi)\right].$$

First, we show that, without loss of generality, we can suppose that all the linear terms of $L(x, \pi)$ are positive.

LEMMA 3.1

$$\text{LD}^* = \underset{\substack{\pi\in\mathbb{R}^n \text{ such that}\\ l_{tp}+\pi_t \geq 0\\ (t=1,\dots,n)\\ (p=1,\dots,m)}}{\text{Max}} \left[\underset{x\in\{0,1\}^{n\,m}}{\text{Min}} L(x,\pi)\right].$$

PROOF. Let $\text{LD}^* = L(x^*, \pi^*)$ where x^* is an optimal solution of

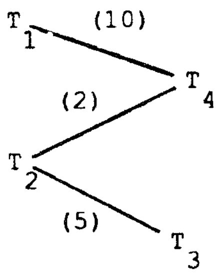
$$\underset{x\in\{0,1\}^{n\,m}}{\text{Min}} L(x,\pi^*)$$

with

$$l_{t_0p_0} + \pi^*_{t_0} < 0.$$

Let

$$\pi = \left(\pi^*_1, \dots, \pi^*_{t_0-1}, -l_{t_0p_0}, \pi^*_{t_0+1}, \dots, \pi^*_n\right).$$



(a)

	P_1	P_2	P_3
T_1	1	2	6
T_2	10	1	2
T_3	10	3	1
T_4	3	4	1

(b)

FIG. 1. (a) Task graph with intertask communication costs. (b) Execution costs

Now, we prove that

$$\mathrm{LD}^* = \operatorname*{Min}_{x \in \{0,1\}^{n\,m}} L(x, \pi).$$

Since $l_{t_0p_0} + \pi^*_{t_0} < 0$ and $c_{tt'} \geq 0$, we get $x^*_{t_0p_0} = 1$. Moreover, since $l_{t_0p_0} + \pi_{t_0} = 0$, we can suppose that $y^*_{t_0p_0} = 1$ for an optimum solution y^* of

$$\operatorname*{Min}_{x \in \{0,1\}^{n\,m}} L(x, \pi).$$

For all x in $\{0, 1\}^{n\,m}$ with

$$x_{t_0p_0} = 1, \qquad L(x, \pi) - L(x, \pi^*) = \sum_{\substack{p=1 \\ p \neq p_0}}^{m} -\left(l_{t_0p_0} + \pi^*_{t_0}\right) x_{t_0p} \geq 0.$$

Then

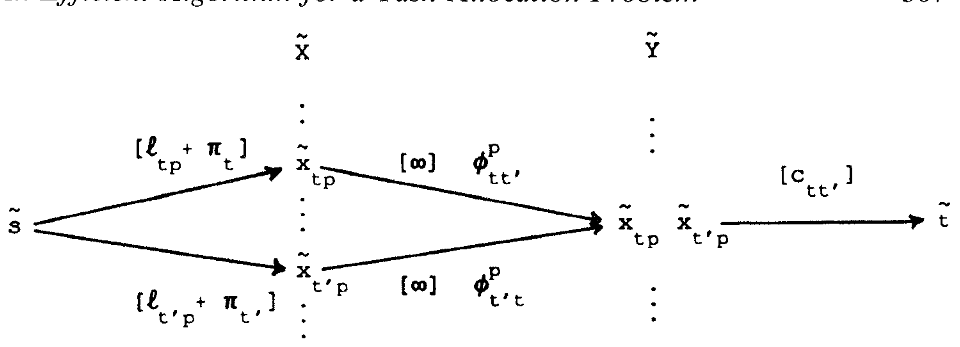
$$\begin{aligned} \operatorname*{Min}_{x \in \{0,1\}^{n\,m}} L(x, \pi) &= \operatorname*{Min}_{\substack{x \in \{0,1\}^{n\,m} \\ \text{such that } x_{t_0p_0} = 1}} L(x, \pi) \\ &\geq \operatorname*{Min}_{\substack{x \in \{0,1\}^{n\,m} \\ \text{such that } x_{t_0p_0} = 1}} L(x, \pi^*) = \mathrm{LD}^*. \end{aligned}$$ □

For any π in $\mathbb{R}^n$, such that $l_{tp} + \pi_t \geq 0$ $(t = 1, \ldots, n)$ and $(p = 1, \ldots, m)$, the Lagrangean function is a "positive-negative" one. It has been proved [14] that minimizing such a function can be performed polynomially via maximum network flow computation. Let us now describe the network R_π associated with the problem

$$\operatorname*{Min}_{x} L(x, \pi).$$

The network R_π is defined as follows: the vertices are the source $\tilde{s}$, the sink $\tilde{t}$ and the two set of vertices $\tilde{X} = \{\tilde{x}_{tp} : t = 1, \ldots, n;\ p = 1, \ldots, m\}$, $\tilde{Y} = \{\tilde{x}_{tp}\tilde{x}_{t'p} : [t, t'] \in U;\ t < t';\ p = 1, \ldots, m\}$.

The flow on the arc $(\tilde{x}_{tp}, \tilde{x}_{tp}\tilde{x}_{t'p})$ (resp., $(\tilde{x}_{t'p}, \tilde{x}_{tp}\tilde{x}_{t'p})$) is denoted by $\phi^p_{tt'}$ (resp., $\phi^p_{t't}$). On Figure 2, the upper-capacity bounds are indicated between



$\tilde{X}$
$\tilde{Y}$
$\tilde{s}$
$[\ell_{tp} + \pi_t]$
$[\ell_{t'p} + \pi_{t'}]$
$\tilde{x}_{tp}$
$\tilde{x}_{t'p}$
$[\infty]$ $\phi^p_{tt'}$
$[\infty]$ $\phi^p_{t't}$
$\tilde{x}_{tp}$ $\tilde{x}_{t'p}$
$[c_{tt'}]$
$\tilde{t}$

FIG. 2. R_π.

square brackets. By using Rhy's results [14], it is easy to prove that

$$\min_{x \in \{0,1\}^{n\,m}} L(x,\pi) = -\sum_{t=1}^{n} \pi_t + V(\phi_\pi) - p \cdot \sum_{\substack{t,t' \text{ such that} \\ [t,t'] \in U, t<t'}} c_{tt'}$$

where ϕ_π is a maximum flow on R_π and $V(\phi_\pi)$ the value of this flow. Moreover, a solution x^* is given by the labeling algorithm of Ford and Fulkerson [8] in the following way: $x^*_{tp} = 0$ if the vertex $\tilde{x}_{tp}$ is labeled $(+)$ or $(-)$, and $x^*_{tp} = 1$, otherwise.

Therefore, the Lagrangean dual problem can be written as:

$$\mathrm{LD}^* = \max_{\substack{\pi \in \mathbb{R}^n \text{ such that } l_{tp} + \pi_t \geq 0 \\ (t=1,\ldots,n) \\ (p=1,\ldots,m) \\ \text{and } \phi_\pi \text{ max flow on } R_\pi}} \left[-\sum_{t=1}^{n} \pi_t + V(\phi_\pi) - p \cdot \sum_{\substack{t,t' \text{ such that} \\ [t,t'] \in U, t<t'}} c_{tt'} \right].$$

This last problem can be written as the following linear programming problem:

$$\left.\begin{array}{lll}
\text{Max} & -\sum_{t=1}^{n} \pi_t + \sum_{\substack{t,t' \text{ such that} \\ [t,t'] \in U, t<t'}} \sum_{p=1}^{m} \left(\phi^p_{tt'} + \phi^p_{t't} - c_{tt'}\right) & \\
\text{subject to} & \phi^p_{tt'} + \phi^p_{t't} \leq c_{tt'} & \\
& \left([t,t'] \in U;\, t < t';\, p = 1,\ldots,m\right) & (3.1.1) \\
& -\pi_t + \sum_{\substack{t' \text{ such that} \\ [t,t'] \in U}} \phi^p_{tt'} \leq l_{tp} & \\
& (t = 1,\ldots,n;\, p = 1,\ldots,m) & (3.1.2) \\
& \phi^p_{tt'} \geq 0,\, \pi_t \in \mathbb{R} & \\
& \left(p = 1,\ldots,m;\, [t,t'] \in U;\, t = 1,\ldots,n\right) &
\end{array}\right\} \quad (3.1)$$

LEMMA 3.2. *There exists an optimum solution to* (3.1) *such that*

$$\phi^p_{tt'} + \phi^p_{t't} = c_{tt'} \ \left([t, t'] \in U;\ t < t';\ p = 1, \dots, m\right).$$

PROOF. Let (ϕ, π) be an optimum solution to (3.1) with $\phi^{p_0}_{t_0t'_0} + \phi^{p_0}_{t'_0t_0} < c_{t_0t'_0}$ and $\epsilon = c_{t_0t'_0} - (\phi^{p_0}_{t_0t'_0} + \phi^{p_0}_{t'_0t_0})$ with $\epsilon > 0$.

Let us consider (φ, π') such that:

$$\pi'_{t_0} = \pi_{t_0} + \epsilon; \qquad \pi'_t = \pi_t \qquad (t \neq t_0;\ t = 1, \dots, n)$$

$$\begin{cases} \varphi^{p_0}_{t_0t'_0} = \phi^{p_0}_{t_0t'_0} + \epsilon; \\ \varphi^{p_0}_{t'_0t_0} = \phi^{p_0}_{t'_0t_0} \end{cases} \qquad \begin{cases} \varphi^p_{tt'} = \phi^p_{tt'} \\ \varphi^p_{t't} = \phi^p_{t't} \end{cases}$$

$$\left([t, t'] \in U;\ t < t';\ p = 1 \cdots m;\ [t, t'] \neq [t_0, t'_0] \text{ or } p \neq p_0\right).$$

It is clear that (φ, π') is an optimum solution to (3.1) with

$$\varphi^{p_0}_{t_0t'_0} + \varphi^{p_0}_{t'_0t_0} = c_{t_0t'_0}. \qquad \square$$

By Lemma 3.2, the linear programming problem (3.1) can be simplified and written as:

$$\left.\begin{array}{lll} \text{Max} & -\sum_{t=1}^{n} \pi_t & \\ \text{subject to} & \phi^p_{tt'} + \phi^p_{t't} = c_{tt'} & \\ & \quad \left([t, t'] \in U;\ t < t';\ p = 1, \dots, m\right) & (3.2.1) \\ & -\pi_t + \sum_{\substack{t' \text{ such that} \\ [t,t'] \in U}} \phi^p_{tt'} \le l_{tp} & \\ & \quad (t = 1, \dots, n;\ p = 1, \dots, m) & (3.2.2) \\ & \phi^p_{tt'} \ge 0,\ \pi_t \in \mathbb{R} & \\ & \quad \left(p = 1, \dots, m;\ [t, t'] \in U;\ t = 1, \dots, n\right) & \end{array}\right\} \quad (3.2)$$

Example (continued). R_π is represented in Figure 3. The optimum solution to (3.2) is given by $\pi = (-5, -5, 0, 1)$ and the flow ϕ which is indicated on the figure (between parentheses). Therefore, the optimum value of the Lagrangean dual is 9 and by Ford and Fulkerson's labeling algorithm, we get $x_{11} = x_{41} = x_{43} = x_{23} = x_{33} = 1$, the other variables being equal to 0. Let us note that this solution is not a primal one.

4. *Approximate Solution to the Lagrangean Dual Problem*

We have seen that the optimum dual solution can be computed as a linear programming problem (see (3.2)). Unfortunately this program involves a huge number of variables and constraints. Indeed for a primal problem with 20 processors ($m = 20$), 50 tasks ($n = 50$), and 200 intertask communications ($|U| = 200$) the program (3.2) includes 8050 variables, 4000 constraints of type (3.2.1), and 1000 constraints of type (3.2.2). Therefore, we have prefered a suboptimal procedure which uses the specificity of R_π. From Lemma 3.2, for

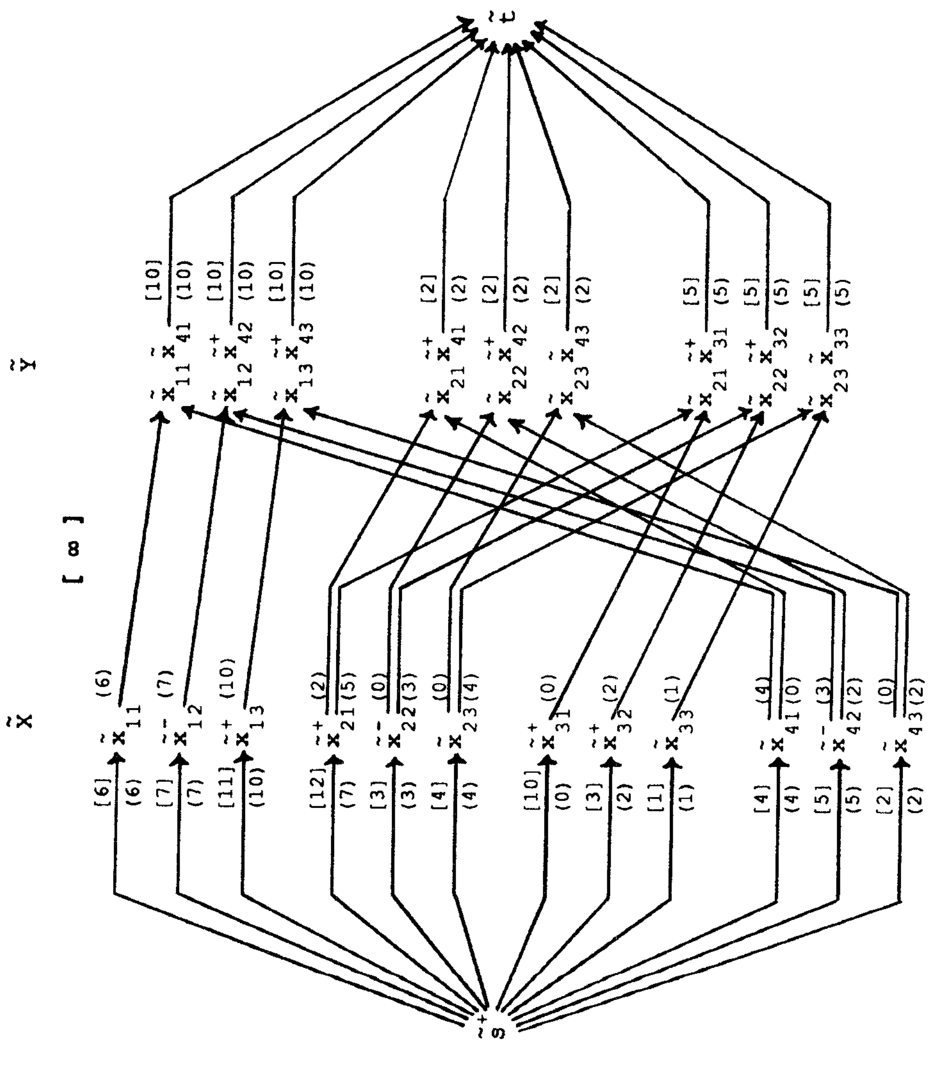


FIG. 3. An example of R_{π}.

any π, we only consider the maximum flow ϕ_π on R_π, which saturates the arcs incident to the sink $\tilde{t}$. Therefore,

$$\operatorname*{Min}_{x \in \{0,1\}^{n\,m}} L(x, \pi) = -\sum_{t=1}^{n} \pi_t$$

and we denote this quantity by $L(\pi)$.

The proof of the following proposition gives a way for constructing $(\phi^k, \pi^k)_{k \in \{1, \ldots, K\}}$ with $L(\pi^{k+1}) > L(\pi^k)$.

PROPOSITION 4.1. *Let ϕ be a max flow on R_π and $\tilde{X}_t = \{\tilde{x}_{tp} : p = 1, \ldots, m\}$. If there exists t_0 in $\{1, \ldots, n\}$ such that all the vertices of $\tilde{X}_{t_0}$ are labeled by the Ford and Fulkerson algorithm, then there exists a dual solution (ϕ', π') such that $L(\pi') > L(\pi)$.*

PROOF. For all p in $\{1, \ldots, m\}$, $\tilde{x}_{t_0 p}$ is labeled by $(+)$ or $(-)$.

Case 1. $\tilde{x}_{t_0 p}$ is labeled by $(+)$. Let ϵ_p be the residual capacity of $(\tilde{s}, \tilde{x}_{t_0 p})$ that is $l_{t_0 p} + \pi_{t_0}$ minus the flow on this arc. We have $\epsilon_p > 0$.

Case 2. $\tilde{x}_{t_0 p}$ is labeled by $(-)$. In this case, one can find a path from $\tilde{s}$ to $\tilde{x}_{t_0 p}$ like it is presented in Figure 4. The interesting flows are indicated between parentheses.

Let us denote

$$\epsilon_p = \operatorname{Min}\left(\left(l_{t_q p} + \pi_{t_q}\right) - \varphi_q, \operatorname*{Min}_{i=0,\ldots,q-1} (\varphi_i)\right).$$

We have $\epsilon_p > 0$. Now let

$$\epsilon = \operatorname*{Min}_{p=1,\ldots,m} (\epsilon_p);$$

we have $\epsilon > 0$. For all vertices of $\tilde{X}_{t_0}$ labeled by $(-)$, the flow ϕ is modified as indicated by Figure 5.

Let ϕ' be the new flow obtained. Note that the set of arcs corresponding to p and p', with $p \neq p'$, are disjoint. Therefore, the modifications that are carried out for all values of p are independent. Then, we put $\pi'_{t_0} = \pi_{t_0} - \epsilon$, $\pi'_t = \pi_t$ $(t \neq t_0)$. It is easy to see that ϕ' is a feasible flow on $R_{\pi'}$. Since ϕ' saturates the arcs incident to $\tilde{t}$, it is a maximum flow on $R_{\pi'}$. Moreover,

$$L(\pi') = -\sum_{t=1}^{n} \pi'_t = \epsilon - \sum_{t=1}^{n} \pi_t = L(\pi) + \epsilon > L(\pi). \qquad \square$$

We only use this proposition for iteratively improving the feasible dual solution. However, when for all $t \in \{1, \ldots, n\}$, $\tilde{X}_t$ includes at least one vertex that is not labeled by $(+)$ or $(-)$, we use a heuristic method to find another feasible dual solution (ϕ', π') with $L(\pi) = L(\pi')$ and such that the number of labeled vertices according to ϕ' on $R_{\pi'}$ is strictly greater than the number of labeled vertices to ϕ on R_π. By iteratively using this heuristic, we hope it will be possible to apply again Proposition 4.1.

5. *Duality Gap*

We are going to show that the coincidence of the primal and dual optimum values can be checked by verifying the existence of a solution to a pseudo-

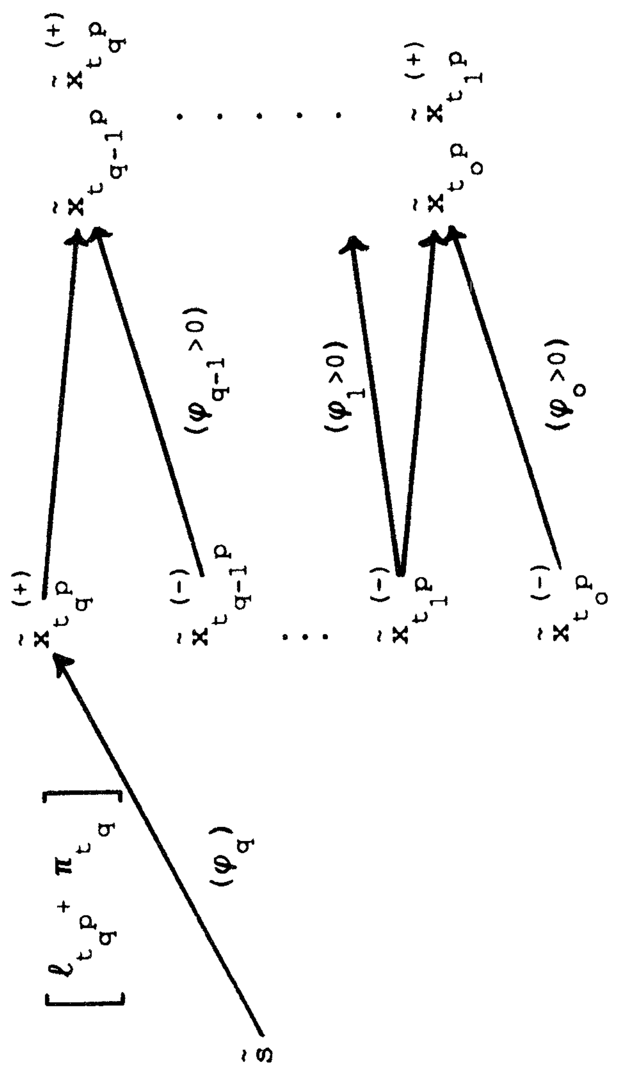
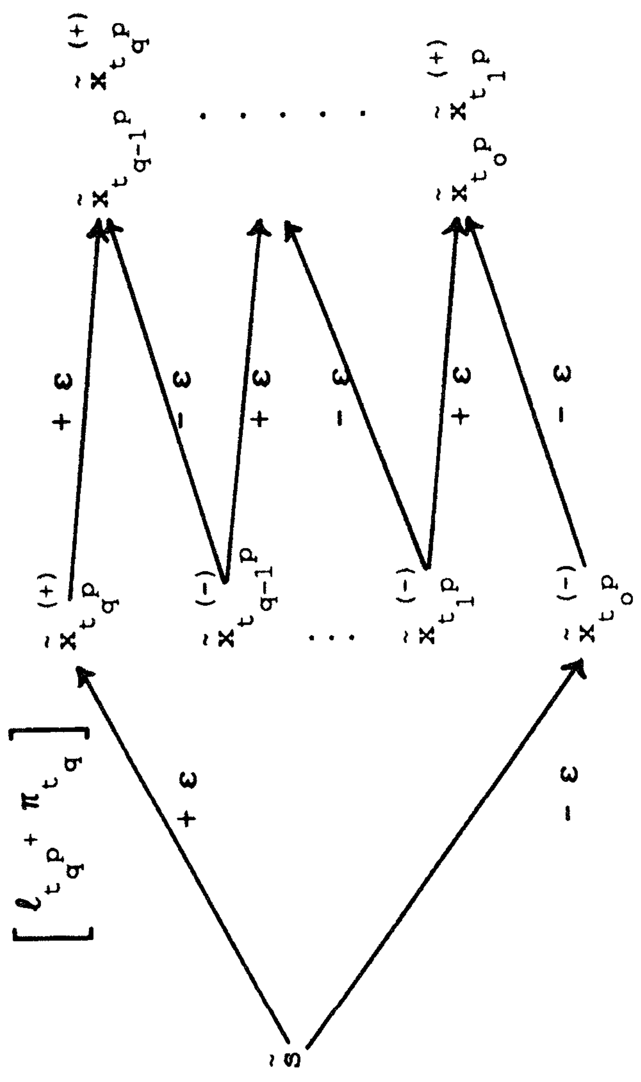

$\tilde{s}$
$[\ell_{t_q p} + \pi_{t_q}]$
(φ_q)
$\tilde{x}_{t_q p}^{(+)}$
$\tilde{x}_{t_{q-1} p}^{(-)}$
$\tilde{x}_{t_1 p}^{(-)}$
$\tilde{x}_{t_o p}^{(-)}$
$(\varphi_{q-1} > 0)$
$(\varphi_1 > 0)$
$(\varphi_o > 0)$
$\tilde{x}_{t_{q-1} p}$
$\tilde{x}_{t_q p}^{(+)}$
$\tilde{x}_{t_o p}$
$\tilde{x}_{t_1 p}^{(+)}$

FIG. 4. An augmenting path on R_π



$$\begin{tikzcd}
 & \tilde{x}^{(+)}_{t_{p_q}} \arrow[r, "\omega^+"] & \tilde{x}_{t_{p_{q-1}}} \quad \tilde{x}^{(+)}_{t_{p_q}} \\
 & \tilde{x}^{(-)}_{t_{p_{q-1}}} \arrow[ur, "\omega^-"] \arrow[r, "\omega^+"] & \cdot \\
\tilde{s} \arrow[uur, "{\omega^+ \left[\ell_{t_{p_q}} + \pi_{t_q}\right]}"] \arrow[ddr, "\omega^-"'] & \cdots & \cdots \\
 & \tilde{x}^{(-)}_{t_{p_1}} \arrow[uur, "\omega^-"] \arrow[r, "\omega^+"] & \tilde{x}_{t_{p_0}} \quad \tilde{x}^{(+)}_{t_{p_1}} \\
 & \tilde{x}^{(-)}_{t_{p_0}} \arrow[ur, "\omega^-"'] &
\end{tikzcd}$$

FIG. 5. Transformation of flow ϕ.

Boolean equation $T(x, \bar{x}) = 0$ where T is a homogeneous posiform. (A similar result is obtained in [1] for unconstrained $0 - 1$ optimization.) If such a solution exists, it gives an optimum solution to the primal problem. Solving the equation $T(x, \bar{x}) = 0$ belongs to the class of the NP-complete problems. But in our particular problem the computational tests have shown that, for practical purposes, T was often a quadratic posiform. That is very interesting because, in this case, the question "Is there a solution to $T(x, \bar{x}) = 0$?" can be answered in a time proportional to the number of terms of T (see [2]).

Classically, for a given dual solution (ϕ, π), there is no duality gap if and only if there exists $x^* \in \{0, 1\}^{n.m}$ such that

$$\text{(i)} \qquad \min_{x \in \{0,1\}^{n\,m}} L(x, \pi) = L(x^*, \pi) = -\sum_{t=1}^{n} \pi_t,$$

$$\text{(ii)} \qquad \sum_{p=1}^{m} x^*_{tp} = 1 \qquad (t = 1, \ldots, n),$$

$$\text{(iii)} \qquad x^*_{tp} \in \{0, 1\} \qquad (t = 1, \ldots, n;\ p = 1, \ldots, m).$$

Moreover, if there exists such an x^*, it is an optimum primal solution.

LEMMA 5.1. *Let (ϕ, π) be a dual solution and r_{tp} be the residual capacity of the arc $(\tilde{s}, \tilde{x}_{tp})$ $(t = 1, \ldots, n;\ p = 1, \ldots, m)$; there is no duality gap if and only if there exists $x \in \{0, 1\}^{n\,m}$ such that $T(x, \bar{x}) = 0$ with*

$$T(x, \bar{x}) = \sum_{\substack{t,p \text{ such that} \\ t=1,\ldots,n; \\ p=1,\ldots,m; \\ r_{tp}>0.}} x_{tp} + \sum_{\substack{t,t' \text{ such that} \\ [t,t'] \in U.}} \sum_{\substack{p \text{ such that} \\ p=1,\ldots,m; \\ \phi^p_{tt'}>0.}} x_{tp}\bar{x}_{t'p}$$

$$+ \sum_{t=1}^{n} \sum_{\substack{p,p' \text{ such that} \\ p'=1,\ldots,m; \\ p=1,\ldots,m; \\ p<p'.}} x_{tp}x_{tp'} + \sum_{t=1}^{n} \prod_{p=1}^{m} \bar{x}_{tp}.$$

PROOF. Since

$$\sum_{p=1}^{m} x_{tp} \le 1 \Leftrightarrow \sum_{p<p'} x_{tp}x_{tp'} = 0 \qquad (t = 1, \ldots, n)$$

and

$$\sum_{p=1}^{m} x_{tp} \ge 1 \Leftrightarrow \prod_{p=1}^{m} \bar{x}_{tp} = 0 \qquad (t = 1, \ldots, n),$$

we have to prove:

$$L(x, \pi) = -\sum_{t=1}^{n} \pi_t \Leftrightarrow \sum_{\substack{t,p \text{ such that} \\ t=1,\ldots,n; \\ p=1,\ldots,m; \\ r_{tp}>0.}} x_{tp} + \sum_{\substack{t,t' \text{ such that} \\ [t,t'] \in U.}} \sum_{\substack{p \text{ such that} \\ p=1,\ldots,m; \\ \phi^p_{tt'}>0.}} x_{tp}\bar{x}_{t'p} = 0.$$

By definition of the Lagrangean function,

$$L(x, \pi) = -\sum_{t=1}^{n} \pi_t \Leftrightarrow \sum_{t=1}^{n}\sum_{p=1}^{m} (l_{tp} + \pi_t) x_{tp} - \sum_{\substack{t,t' \text{ such that} \\ [t,t'] \in U; \\ t<t'}} \sum_{p=1}^{m} c_{tt'} x_{tp} x_{t'p} = 0.$$

(ϕ, π) being a solution to (3.2), we have:

$$c_{tt'} = \phi_{tt'}^{p} + \phi_{t't}^{p} \qquad ([t, t'] \in U; t < t'; p = 1, \dots, m).$$

Therefore, we get:

$$L(x, \pi) = -\sum_{t=1}^{n} \pi_t \Leftrightarrow \sum_{t=1}^{n}\sum_{p=1}^{m} (l_{tp} + \pi_t) x_{tp} - \sum_{\substack{t,t' \text{ such that} \\ [t,t'] \in U.}} \sum_{p=1}^{m} \phi_{tt'}^{p} x_{tp} x_{t'p} = 0,$$

and by substituting $1 - \bar{x}_{t'p}$ to $x_{t'p}$, it follows:

$$L(x, \pi) = -\sum_{t=1}^{n} \pi_t \Leftrightarrow \sum_{t=1}^{n}\sum_{p=1}^{m} \left[(l_{tp} + \pi_t) - \sum_{\substack{t' \text{ such that} \\ [t,t'] \in U}} \phi_{tt'}^{p} \right] x_{tp} + \sum_{\substack{t,t' \text{ such that} \\ [t,t'] \in U}} \sum_{p=1}^{m} \phi_{tt'}^{p} x_{tp} \bar{x}_{t'p} = 0.$$

Since, by definition,

$$(l_{tp} + \pi_t) - \sum_{\substack{t' \text{ such that} \\ [t,t'] \in U.}} \phi_{tt'}^{p} = r_{tp},$$

we get finally:

$$L(x, \pi) = -\sum_{t=1}^{n} \pi_t \Leftrightarrow \sum_{\substack{t,p \text{ such that} \\ t=1,\dots,n; \\ p=1,\dots,m; \\ r_{tp}>0}} x_{tp} + \sum_{\substack{t,t' \text{ such that} \\ [t,t'] \in U}} \sum_{\substack{p \text{ such that} \\ p=1,\dots,m; \\ \phi_{tt'}^{p}>0}} x_{tp} \bar{x}_{t'p} = 0. \qquad \square$$

Notice that some variables must be equal to 0 in all solutions to $T(x, \bar{x}) = 0$. Let F_0 be the set of these variables. F_0 is constructed from $T(x, \bar{x})$ in the following way:

— $F_0 \leftarrow \{x_{tp} : r_{tp} > 0\}$, that is, $F_0 \leftarrow \{x_{tp} : \tilde{x}_{tp}$ is labeled $(+)$ for ϕ on $R_\pi\}$;
— If $x_{t'p} \in F_0$ and $\phi_{tt'}^p > 0$, then $F_0 \leftarrow F_0 \cup \{x_{tp} : [t, t'] \in U;\ p = 1, \dots, m\}$.

Notice that this second rule corresponds exactly to the labeling of the vertices of $\tilde{X}$ by $(-)$ (see Figure 2 and [8]). Therefore,

$$F_0 = \{x_{tp} : \tilde{x}_{tp} \text{ is labeled } (+) \text{ or } (-) \text{ for } \phi \text{ on } R_\pi\}.$$

We are now in position to state the following theorem for characterizing the existence of a duality gap.

THEOREM 5.2. *Let (ϕ, π) be a dual solution and $N_t \subseteq \{1, \dots, m\}$ defined by $N_t = \{p \in \{1, \dots, m\} : \tilde{x}_{tp}$ is not labeled $(+)$ or $(-)$ for ϕ on $R_\pi\}$, $(t = 1, \dots, n)$. There is no duality gap if and only if it exists x in $\{0, 1\}^{n.m}$ such that $T'(x, \bar{x}) = 0$ with*:

$$T'(x, \bar{x}) = \sum_{\substack{t,t' \text{ such that} \\ [t,t'] \in U}} \ \sum_{\substack{p \text{ such that} \\ p \in N_t \cap N_{t'}, \\ \phi_{tt'}^p > 0.}} x_{tp}\bar{x}_{t'p}$$
$$+ \sum_{t=1}^{n} \ \sum_{\substack{p,p' \text{ such that} \\ p \in N_t;\, p' \in N_t; \\ \text{and } p < p'.}} x_{tp}x_{tp'} + \sum_{t=1}^{n} \prod_{p \in N_t} \bar{x}_{tp}.$$

Solving the equation $T(x, \bar{x}) = 0$ is an NP-complete problem since T always includes the terms

$$\prod_{p=1}^{m} \bar{x}_{tp} \qquad (t = 1, \dots, n)$$

of degree m. But in $T'(x, \bar{x})$, the terms

$$\prod_{p \in N_t} \bar{x}_{tp} \qquad (t = 1, \dots, n)$$

are nonquadratic only if $|N_t| > 2$. The computational experiments (see next section) have shown that this fact was exceptional. So the question "Is there a duality gap?" will be often answered by a polynomial-time algorithm (linear in number of terms in T'). In Section 4, we used Ford and Fulkerson's labeling rules to compute an approximate solution to the Lagrangean dual problem. Again, note the importance of these rules to obtain an efficient characterization of a zero gap.

Example (continued). We get

$$N_1 = \{x_{11}\}; \qquad N_2 = \{x_{23}\}; \qquad N_3 = \{x_{33}\}; \qquad N_4 = \{x_{41}, x_{43}\}$$

and

$$T'(x, \bar{x}) = \bar{x}_{11} + \bar{x}_{23} + \bar{x}_{33} + \bar{x}_{41}\bar{x}_{43}$$
$$+x_{41}x_{43} + x_{11}\bar{x}_{41} + x_{41}\bar{x}_{11}$$
$$+x_{23}\bar{x}_{33} + x_{33}\bar{x}_{23} + x_{43}\bar{x}_{23}.$$

The quadratic equation $T'(x, \bar{x}) = 0$ admits the solution

$$x^*_{11} = x^*_{23} = x^*_{33} = x^*_{41} = 1,$$

the other variables being equal to 0. Therefore, there is no duality gap, and x^* is an optimum primal solution; its value is 9.

6. *Algorithm and Computational Results*

We use a branch-and-bound algorithm, with best first-branch method, to solve the primal problem.

The following evaluation and test method shall be used for each node:

— Compute an approximate dual solution (see Section 4);
— Is there a zero gap or is there a solution to $T'(x, \bar{x}) = 0$? (see Section 5);
— If there is a zero gap, then stop, and x^*, solution to $T' = 0$, is an optimum primal solution for the considered subproblem.

The following branching shall be used: When there is no solution to $T'(x, \bar{x}) = 0$, the considered subproblem is divided into several subproblems as follows:

Let $t_0 = \text{Max}_{t=1,\ldots,n}(|N_t|)$ and $\{x_{t_0p} : p \in N_{t_0}\} = \{x_{t_0p_1}, \ldots, x_{t_0p_q}\}$.

We have $q + 1$ subproblems.

Subproblem 1: $x_{t_0p_1} = 1$;

Subproblem q: $x_{t_0p_q} = 1$;

Subproblem $q + 1$: $x_{t_0p_1} = x_{t_0p_2} = \cdots = x_{t_0p_q} = 0$.

Our algorithm was coded in Pascal and run on a DEC VAX-780. The instances are randomly generated. The execution costs are between 1 and B_{proc} and the communication costs between 1 and B_{com}. Values of B_{proc} and B_{com} were chosen in such a way that the sum of the execution costs is comparable with the sum of the communication costs. So, intuitively, an optimum solution is more difficult to find. For each considered size, 20 instances are generated; the resuls are shown in Table I.

Table II gives the worst-case results (for the C.P.U. time) on the 20 instances.

7. *Concluding Remarks*

For one instance with $n = 10$, $m = 101$, and $|U| = 307$, the optimum was not obtained after 1 hour of C.P.U. time usage and in this case we only have a very good approximate solution (less than 0.5% from the optimum). This instance has not been considered for the presentation of the results in Tables I and II.

TABLE I. RESULTS OF ALGORITHM (20 INSTANCES)

m	n	$\lvert U \rvert$	B_{com}	B_{proc}	Err	Root	Nb_1	Nb_2	Nb_3	Nb_4	Nb. Eval	CPU (seconds)
10	29	61	23	101	0%	19	25.85	3.1	0.05	0	1.15	17
15	37	113	17	101	0%	14	33.65	3.3	0.15	0	1.85	52
20	53	229	11	101	0%	11	45.75	6.9	0.35	0	2.50	171
10	101	307	29	101	0.2%	10	88.8	10.5	1	0.7	7.35	876

NOTE: m = number of processors; n = number of tasks; $\lvert U \rvert$ = number of intertask communications; B_{com} = the communication costs belong to $[1, B_{\mathrm{com}}]$; B_{proc} = the execution costs belong to $[1, B_{\mathrm{proc}}]$; Err = relative error between the optimum primal solution and the approximate dual solution at the root of the tree (mean value on the 20 instances); Root = number of instances such that the approximate dual solution is an optimum dual solution, and there is no duality gap at the root of the tree; Nb_k ($k = 1, 2, 3, 4$) = number of set N_t ($t = 1, \ldots, n$) such that $k = \lvert N_t \rvert$ (mean value on the 20 instances); Nb. Eval = number of nodes that are evaluated in the branch-and-bound algorithm (mean value on the 20 instances); CPU = C.P.U. time in seconds (mean value on the 20 instances).

TABLE II. WORST-CASE RESULTS (FOR CPU TIME) (20 INSTANCES)

m	n	$\lvert U \rvert$	B_{com}	B_{proc}	Eval	Sol	Nb_1	Nb_2	Nb_3	Nb_4	Nb. Eval	CPU (seconds)
10	29	61	23	101	765	766	29	0	0	0	3	27
15	37	113	17	101	1001	1005	33	4	0	0	5	118
20	53	229	11	101	1454	1458	46	7	0	0	11	377
10	101	307	29	101	3569	3583	78	23	0	0	55	3083

NOTE: m = number of processors; n = number of tasks; $\lvert U \rvert$ = number of intertask communications; B_{com} = the communication costs belong to $[1, B_{\mathrm{com}}]$; B_{proc} = the execution costs belong to $[1, B_{\mathrm{proc}}]$; Eval = value of the approximate dual solution at the root of the tree; Sol = value of the optimum primal solution; Nb_k ($k = 1, 2, 3, 4$) = number of set N_t ($t = 1, \ldots, n$) such that $k = \lvert N_t \rvert$; Nb. Eval = number of nodes that are evaluated in the branch-and-bound algorithm; CPU = C.P.U. time in seconds.

These tables show that the bound given by an approximate dual solution is excellent and that the relative error is always less than 0.4%. This can be explained by the weak frequency of duality gaps and by the ability of our suboptimal procedure to find frequently an optimum dual solution. We also can remark that the number of terms of degree greater than 3 in $T'(x, \bar{x})$ is very small (see the values of Nb_k). So, practically, the existence of a solution to $T'(x, \bar{x}) = 0$ can be checked in linear time.

ACKNOWLEDGMENTS. The authors wish to thank Ingeneer C. Ikhlef of Centre National d'Etudes des Télécommunications for careful readings of the manuscript, as well as the referees, for numerous and helpful suggestions and remarks.

REFERENCES

1. ADAMS, W. P., BILLIONNET, A., AND SUTTER, A. Unconstrained 0–1 optimization and Lagrangean relaxation. *Disc. Appl. Math. 29* (1990), pp. 131–142.
2. ASPVALL, B., PLASS, M. F., AND TARJAN, R. E. A linear-time algorithm for testing the truth of certain quantified Boolean formulas. *Inf. Proc. Lett. 8* (1979), 121–123.
3. BILLIONNET, A., COSTA, M. C., AND SUTTER, A. Les problèmes de placement dans les systèmes distribués. *Tech. Sci. Inf. 8*, 3 (1989), 307–337.

4. BOKHARI, S. H. Dual processor scheduling with dynamic reassignment *IEEE Trans. Softw. Eng. SE-5*, 4 (July 1979), 341–349.
5. BOKHARI, S. H. On the mapping problem. *IEEE Trans. Comput. C-30* (Mar 1981), 207–214.
6. CHU, W. W., HOLLOWAY, L. J., LAN, M., AND EFE, K. Task allocation in distributed data processing *Comput. 13*, 11 (Nov. 1980), 57–69.
7. EFE, K. Heuristic models of task assignment scheduling in distributed systems. *Comput. 15* (June 1982), 50–56.
8. FORD, L. R., AND FULKERSON, D. R. *Flows in Networks*. Princeton University Press, Princeton, N.J., 1962.
9. GABRIELIAN, A., AND TYLER, D. B. Optimal object allocation in distributed computing system In *Proceedings of the International Conference on Distributed Computing Systems* (San Francisco, Calif., May 14–18). 1984, pp. 88–95.
10. LO, V. M. Heuristic algorithms for task assignment in distributed systems. In *Proceedings of the International Conference on Distributed Computing Systems* (San Francisco, Calif., May 14–18). 1984, pp. 30–39.
11. MA, P R., LEE, E. Y S., AND TSUCHIGA, M. A task allocation model for distributed computing systems. *IEEE Trans. Comput. C-31*, 1 (Jan 1982), 41–47.
12. PRICE, C. C., AND KRISHNAPRASAD, S. Software allocation models for distributed computing systems. In *Proceedings of the International Conference on Distributed Computing Systems* (San Francisco, Calif., May 14–18). 1984, pp. 40–48.
13. RAO, G. S., STONE, H. S., AND HU, T. C. Assignment of tasks in a distributed processor system with limited memory. *IEEE Trans. Comput. C-28*, 4 (Apr. 1979), 291–299.
14. RHYS, J. A selection problem of shared fixed costs and networks. *Manage. Sci. 17* (1970), 200–207.
15. SHEN, C., AND TSAI, W. A graph-matching approach to optimal task assignment in distributed computing systems using a minimax criterion. *IEEE Trans. Comput. C-34*, 3 (Mar. 1985), 197–203.
16. SINCLAIR, J. B. Efficient computation of optimal assignments for distributed tasks. *J. Paral. Dist. Comput. 4* (1987), 342–362.
17. STONE, H. S. Multiprocessor scheduling with the aid of network flow algorithms. *IEEE Trans. Softw. Eng. SE-3* (Jan. 1977), 85–93.

RECEIVED JUNE 1988; REVISED NOVEMBER 1989 AND OCTOBER 1990; ACCEPTED OCTOBER 1990