Job Shop Using Lagrangian Relaxation

Haoxun Chen, Chengbin Chu, Jean-Marie Proth

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Job Shop Scheduling Using Lagrangian Relaxation

Haoxun CHEN - Chengbin CHU - Jean-Marie PROTH

N° 2505
Mars 1995

PROGRAMME 5

Job Shop Scheduling Using Lagrangian Relaxation

Haoxun Chen
Xi'an Jiaotong University, Xi'an, People's Republic of China and
visiting professor at INRIA-Lorraine, CESCOM Technopôle Metz 2 000.
4 rue Marconi, 57070 Metz, France

Chengbin Chu
INRIA-Lorraine, CESCOM Technopôle Metz 2 000.
4 rue Marconi, 57070 Metz, France

Jean-Marie Proth
INRIA-Lorraine, CESCOM Technopôle Metz 2 000,
4 rue Marconi, 57070 Metz, France and
Institute for Systems Research, University of Maryland,
College Park, MD, USA

Abstract. Lagrangian relaxation has recently emerged as an important method for solving complex scheduling problems. The technique has succeesfully been used to obtain near-optimal solutions for one machine scheduling problems and parallel machine scheduling problems. The approach consists of relaxing the capacity constraints on machines by using Lagrangian multipliers. The relaxed problem can be decomposed into independent job level subproblems. Peter B. Luh and his colleagues extended the technique to general job shop scheduling problems by introducing more Lagrangian multipliers to relax the precedence constraints among operations, such that each job level relaxed subproblem can be further decomposed into a set of operation level subproblems which can easily be solved by enumeration. Unfortunately, the operation level subproblems exhibit solution oscillation from iteration to iteration and, in many cases, prevent convergence of the algorithm. Although they have proposed several methods to prevent solution from oscillation, none of the methods is really satisfactory. In this paper, we propose an efficient pseudopolynomial time dynamic programming algorithm to relaxed job level subproblems. We show that by extending the technique to job shop scheduling problems, the relaxation of the precedence constraints is unnecessary, and thus the oscillation problem vanishes. This algorithm also results in a much more efficient Lagrangian relaxation approach to job-shop scheduling problems. Furthermore, this algorithm makes it possible to optimize "min-max" criteria by Lagrangian relaxation. These criteria have been neglected in Lagrangian relaxation litterature for sake of indecomposability. Computational results on randomly generated problems are given to demonstrate the efficiency of the algorithm.

Keywords. Lagrangian Relaxation, Scheduling, Decomposition, Dynamic Programming.

Ordonnancement des "job shops" avec la relaxation Lagrangienne

Haoxun Chen
Xi'an Jiaotong University, Xi'an, People's Republic of China et
professeur invité à l'INRIA-Lorraine, CESCOM Technopole Metz 2 000,
4 rue Marconi, 57070 Metz, France

Chengbin Chu
INRIA-Lorraine, CESCOM Technopôle Metz 2 000,
4 rue Marconi, 57070 Metz, France

Jean-Marie Proth
INRIA-Lorraine, CESCOM Technopôle Metz 2 000,
4 rue Marconi, 57070 Metz, France et
Institute for Systems Research, University of Maryland,
College Park, MD, USA

Résumé. Depuis quelques années, la relaxation Lagrangienne est devenue une méthode importante pour résoudre des problèmes d'ordonnancement complexes. Cette technique a été appliquée avec succès aux problèmes d'ordonnancement à une machine et aux problèmes d'ordonnancement en machines parallèles, pour obtenir des solutions approchées. Cette approche consiste à relaxer les contraintes de capacité à l'aide de multiplicateurs Lagrangiens. Le problème ainsi relaxé peut être décomposé en sous-problèmes de niveau tâche indépendants. Le Professeur P.B. Luh et ses collègues ont élargi cette technique aux problèmes d'ordonnancement des "job shops" en général, en introduisant des multiplicateurs Lagrangiens supplémentaires pour relaxer les contraintes de précédence entre les opérations d'une tâche, de telle sorte que chaque sous-problème de niveau tâche peut être encore décomposé en sous-problèmes de niveau opération qui peuvent être facilement résolus. Malheureusement, les sous-problèmes de niveau opération conduisent à une oscillation de solutions d'une itération à une autre, et empêchent souvent la convergence de l'algorithme. Bien que plusieurs méthodes aient été proposées pour éviter cette oscillation, aucune d'entre elles ne fournit des résultats concluants. Dans cet article, nous proposons an algorithme pseudopolynômial efficace basé sur la programmation dynamique pour résoudre les sous-problèmes de niveau tâche. Nous montrons qu'en appliquant cet algorithme à des problèmes d'ordonnancement des "job shops", la relaxtion des contraintes de précédence devient redondante, et par la suite, les phénomènes d'oscillation de solution disparaîssent. Cet algorithme rend donc l'approche de la relaxation Lagrangienne beaucoup plus efficace pour résoudre des problèmes d'ordonnancement des "job shops". Par ailleurs, cet algorithme permet d'optimiser des critères de type "min-max" par la méthode de relaxation Lagrangienne, des critères non considérés dans la littérature jusqu'à présent à cause de la non décomposabilité. Les résultats numériques sur des exemples générés aléatoirement sont fournis pour illustrer l'efficacité de l'algorithme.

Mots clés. Relaxation Lagrangienne, Ordonnancement, Décomposition, Programmation dynamique.

1. Introduction

Scheduling is one of the most important issues in the planning and operation of manufacturing systems, but most scheduling problems are NP-hard. The development of optimal polynomial algorithms is unlikely. To solve the problems, many methods have been proposed. These methods can be classified into deterministic optimization methods, approximation/heuristic methods and stochastic optimization methods [1-15].

Deterministic optimization methods include dynamic programming methods [3-5] and branch-and-bound methods [6-11]. Because most of them are based on implicit enumeration of possible schedules, they become computationally expensive even for small size job shops [1-2].

Approximation/heuristic methods usually generate satisfactory schedules in a reasonable computation time, but it is generally very difficult to evaluate how far these schedules are from the optimal solutions [12-13].

Stochastic optimization methods (such as simulated annealing and the genetic algorithms) present a new and promising direction to solve scheduling problems, but as in most other optimization methods, there is some degree of enumeration. Therefore the methods converge to optimal schedules very slowly in general. Similar to most heuristic methods, it is difficult to determine if an optimal schedule has been obtained [14-15] and when.

Lagrangian relaxation (LR) has recently emerged as an important method for solving complex scheduling problems. The technique has successfully been used to obtain near-optimal solutions for one machine scheduling problems and parallel machine scheduling problems [16]. In this mentioned paer, the capacity constraints on a machine or on a set of identical machines are relaxed by using Lagrangian multipliers. Peter B. Luh and his colleagues extended the technique to general job shop scheduling problems by introducing more Lagrange multipliers to relax the precedence constraints among operations, so that each job level relaxed subproblem can be further decomposed into a set of operation level subproblems which can easily be solved by enumeration. Unfortunately, the operation level subproblems exhibit solution oscillation from iteration to iteration and, in many cases, prevent convergence of the algorithm [17-19].

In order to avoid the solution oscillation, Hoitomt and Luh [18-19] propose an augmented LR approach in which some quadratic penalty terms are appended to its Lagrange function. However, the quadratic penalties prevent the corresponding job level relaxed problems from being directly decomposed into operation level subproblems. Czerwinski and Luh [20] proposed an improved LR approach by introducing an auxiliary objective function in which earliness and tardiness penalties for each operation are directly included. Unfortunately, the weighted quadratic tardiness function originally as the cost function of the scheduling problems is modified, and some parameters in the auxiliary objective function such as due dates and starting times for operations are difficult to be taken into account properly. Moreover, the approach does not completely eliminate the solution oscillation. Chen [21] proposed a sequential LR approach which can reduce the solution oscillation to a great extent, but it is at the expense of the increase in computation time. Tomastik and Luh [22] propose a facet ascending algorithm for maximizing the Lagrangian dual of the integer programming problems at hand, which is claimed to be applicable to preventing the solution oscillation. However, only a simple example consisting of scheduling six single operation jobs on two identical machines is given to illustrate the algorithm. No convergence proof and computational complexity analysis is done.

The motivations of decomposing each relaxed job level subproblem into operation level subproblems by relaxing the precedence constraints in [17-19] come from the observation that solving the relaxed job level suproblem by enumerating the starting time of each operation of corresponding job

is impractical if the job has more than three operations since no efficient algorithm for relaxed job level subproblems is available. In this paper, such a situation is changed. Motivated by the work of Adams *et al.* ([13]) on the shifting bottleneck procedure for job shop scheduling, and the work of Hall *et al.* ([5])on dynamic programming algorithms for earliness-tardiness scheduling problems, we propose an efficient pseudopolynomial time dynamic programming algorithm for relaxed job level subproblems. We show that, by extending the Lagrangian relaxation technique to job shop scheduling problems, the relaxation of the precedence constraints is unnecessary. This algorithm also results in a much more efficient Lagrangian relaxation approach to job-shop scheduling problems, in which the solution oscillation disappears. Computational results on randomly generated problems are given to demonstrate the efficiency of the algorithm.

Another important contribution of this work is that our algorithm makes it possible to consider some "min-max" criteria such as makespan. Unitl now, only "min-sum" criteria are considered in the literature, since otherwise the problem cannot be decomposed into job level subproblems. With our algorithm, the problem can still be decomposed into job level subproblems. Only very little additional effort is needed to connect the job level subproblems to global problem.

The remainder of the paper is organized as follows. Section 2 presents a mathematical formulation for job shop scheduling problems with "min-sum" criteria. Section 3 describes a Lagrangian relaxation framework. The dynamic programming algorithm for relaxed job level subproblems is developed under chain/join like precedence constraints is presented in Section 4. Section 5 extends the approach to "min-max" criteria. Finally, numerical experiment presented in Section 6 demonstrate the efficiency of the algorithm and the potential of the new Lagrangian relaxation approach to schedule job shops of realistic sizes.

2. Problem formulation

The problem considered in this paper consists of scheduling N jobs on M machines in order to minimize a weighted quadratic tardiness cost function of the jobs. The completion of each job requires a set of operations, and each operation can be performed on one of a set of machines. Jobs have different due dates and may have different weights. Following variables will be used for the formulation of the scheduling problem, where operation j of job i is referred to as operation (i,j).

N	Number of jobs;
n_i	Number of operations in job i;
M	Number of machines;
C_i	Completion time of job i;
$\mathcal{M}_{i,j}$	Set of machines capable of performing operation (i,j);
$P_{i,j}$	Set of operations of job i immediately preceding operation (i,j);
$t_\mu^{i,j}$	Processing time of operation (i,j) on machine $\mu \in \mathcal{M}_{i,j}$;
$c_{i,j}$	Completion time of operation (i,j);
$m_{i,j}$	Machine selected to process operation (i,j), $m_{i,j} \in \mathcal{M}_{i,j}$;
H	Time horizon under consideration;
J	Objective function to be optimized.

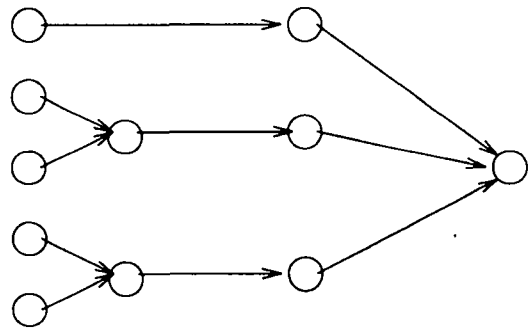

Figure 1: Chain/Join Structured Precedence Constraints

It is assumed that the precedence constraints of a job form a directed acyclic graph and, without loss of generality, that each job ends with a single operation (i, n_i). Therefore, $C_i = c_{i,n_i}$. The processing of an operation is assumed to be nonpreemptive. The time horizon H is assumed to be long enough to complete all the jobs. All jobs are assumed available for processing at time 0 and, for simplicity, all machines are assumed to be in working order throughout the whole time horizon $[0, H]$. Note that all results of this paper can be applied to the situation where some machines are not at working order during some time periods for maintenance purpose, for instance.

The decison variables of the scheduling problem are the starting times of operations $\{c_{i,j}\}$ and the machines $\{m_{i,j}\}$ selected to perform these operations. Each job i is assocaited with a cost function $\varphi_i(C_i)$. The objective function to be minimized is to minimize the total cost $J = \sum_{i=1}^{N} \varphi_i(C_i)$.

Let $\mathbf{1}(\tau) = \begin{cases} 1, & \text{if } \tau \geq 0; \\ 0, & \text{otherwise.} \end{cases}$ Let also $O_\mu = \{(i,j) | m_{i,j} = \mu\}$, the set of operations to be performed on machine μ. The deterministic scheduling problem, denoted by **Problem P** can be formulated as follows:

$$\underset{\{m_{i,j}\},\{c_{i,j}\}}{\text{Minimize}} \quad J = \sum_{i=1}^{N} \varphi_i(C_i) \tag{1}$$

subject to precedence constraints,

$$c_{i,k} \leq c_{i,j} - t^{i,j}_{m_{i,j}}, \quad \forall k \in P_{i,j}, \quad \forall 1 \leq j \leq n_i, \quad \forall 1 \leq i \leq N, \tag{2}$$

and to capacity constraints,

$$\sum_{(i,j) \in O_\mu} \{\mathbf{1}(\tau - c_{i,j} + t^{i,j}_\mu) - \mathbf{1}(\tau - c_{i,j})\} \leq 1, \quad \forall 1 \leq \mu \leq M, \quad \forall 0 \leq \tau \leq H. \tag{3}$$

Constraints (2) and (3) reflects the fact that operation (i, j) can be processed on several machines with different processing time requirements (i.e., simple routing for operations). The processing time requirements are implicitly taken into account in these constraints.

In most of the manufacturing/assembling job shops, typical precedence constraints of a job have chain/join structure as shown in Fig. 1. In this figure, each operation is represented by a node (a circle). A node with more than one immediate predecessor (called "joining") represents an assembly operation. It should be noticed that chain like precedence constraints constitute a special case of chain/join like precedence constraints.

Therefore, in the remainder of this paper, we particularly consider the job shop scheduling problems with chain/join structureed precedence constraints.

3. A Lagrangian Relaxation Framework for Scheduling Problems

Similarly to the approaches proposed in [17–19], we relax the capacity constraints by using Lagrange multipliers. We then decompose the problem into job-level subproblems. However, we do not further decompose the job-level subproblems into operation-level subproblems, since it would result in solution oscillation phenomena as pointed out in the introduction.

3.1. The Lagrangian Relaxation Framework

The capacity constraints (3) can be relaxed by using nonnegative Lagrangian multipliers $\lambda_{\tau,\mu}$ ($\tau = 0, 1, \ldots, H; \mu = 1, 2, \ldots, M$). This leads to the following relaxed problem, denoted by **RP**:

$$\underset{\{m_{i,j}\},\{c_{i,j}\}}{\text{Minimize}} \sum_{i=1}^{N} \left\{ \varphi_i(C_i) + \sum_{\mu=1}^{M} \sum_{\tau=0}^{H} \lambda_{\tau,\mu} \left[\sum_{(i,j)\in O_\mu} (\mathbf{1}(\tau - c_{i,j}) - \mathbf{1}(\tau - c_{i,j} - t_\mu^{i,j})) - 1 \right] \right\} \tag{4}$$

or

$$\underset{\{m_{i,j}\},\{c_{i,j}\}}{\text{Minimize}} \left\{ \sum_{i=1}^{N} \left[\varphi_i(C_i) + \sum_{j=1}^{n_i} \sum_{\tau=c_{i,j}-t_{m_{i,j}}^{i,j}}^{c_{i,j}-1} \lambda_{\tau,m_{i,j}} \right] - \sum_{\mu=1}^{M} \sum_{\tau=0}^{H} \lambda_{\tau,\mu} \right\} \tag{5}$$

subject to (2).

The objective function of Problem **RP** can be rewritten as

$$\sum_{i=1}^{N} \underset{\{m_{i,j}\},\{c_{i,j}\}}{\text{Minimize}} \left\{ \varphi_i(C_i) + \sum_{j=1}^{n_i} \sum_{\tau=c_{i,j}-t_{m_{i,j}}^{i,j}}^{c_{i,j}-1} \lambda_{\tau,m_{i,j}} \right\} - \sum_{\mu=1}^{M} \sum_{\tau=0}^{H} \lambda_{\tau,\mu}. \tag{6}$$

where the minimization operation in (4) has been brought inside the summation since the jobs are independent.

The Lagrangian dual to problem P, denoted by **DP**, is

$$\underset{\{\lambda_{\tau,\mu}\geq 0\}}{\text{Maximize}}\ L = -\sum_{\mu=1}^{M} \sum_{\tau=0}^{H} \lambda_{\tau,\mu} + \sum_{i=1}^{N} \min_{\{m_{i,j}\},\{c_{i,j}\}} \left\{ \varphi_i(C_i) + \sum_{j=1}^{n_i} \sum_{\tau=c_{i,j}-t_{m_{i,j}}^{i,j}}^{c_{i,j}-1} \lambda_{\tau,m_{i,j}} \right\} \tag{7}$$

The objective function (6) results in a decomposition framework in which a minimization subproblem for each job i, denoted by $\mathbf{SP_i}$, has to be solved:

$$\underset{\{m_{i,j}\},\{c_{i,j}\}}{\text{Minimize}} \left\{ \varphi_i(C_i) + \sum_{j=1}^{n_i} \sum_{\tau=c_{i,j}-t_{m_{i,j}}^{i,j}}^{c_{i,j}-1} \lambda_{\tau,m_{i,j}} \right\} \tag{8}$$

This decomposition leads to an approach to solve the scheduling problem. As in [16-19], there are several steps to obtain a near-optimal solution, namely: solve subproblems, solve the dual problem, and construct a feasible solution. The effectiveness of the decomposition framework depends on the effectiveness of the algorithm to solve the subproblems. In the next section, we propose an efficient pseudopolynomial dynamic programming algorithm to solve job-level subproblems. The rest of this section is devoted to a brief description of the solution of the dual problem and the construction of a feasible solution.

3.2. Solving the dual problem

The dual problem can be solved by using the subgradient method [23]. The Lagrange multipliers $\{\lambda_{\tau,\mu}\}$ which constitute a vector λ are updated using the follwing formula

$$\lambda^{(h+1)} = \lambda^{(h)} + \alpha^{(h)} g(\lambda^{(h)}) \tag{9}$$

where $\lambda^{(h)}$, $\alpha^{(h)}$ are respectively the value of vector λ obtained at the hth iteration, and the step size at the hth iteration. $g(\lambda)$ is the subgradient of the dual function (7) which is a vector composed of $g_{\tau,\mu}(\lambda)$'s, where $g_{\tau,\mu}(\lambda) = \sum_{(i,j)\in O_\mu} \left\{ \mathbf{1}(\tau - c_{i,j}) - \mathbf{1}(\tau - c_{i,j} - t_\mu^{i,j}) \right\} - 1$. The step size is given by

$$\alpha^{(h)} = \beta \times \frac{L^U - L^{(h)}}{(g(\lambda^{(h)}))^T g(\lambda^{(h)})}, \tag{10}$$

where L^U is an upper bound of the optimal objective value of (7). $L^{(h)}$ is the value of the dual function at the hth iteration, and $0 < \beta < 2$. The superscript "T" indicates the transposition operation.

The convergence of this subgradient algorithm depends on the selection of parameters β, L^U and $\lambda^{(0)}$. It has been shown that the adaptive change of the parameters β, L^U with the progress of the subgradient algorithm will speed up the convergence of the algorithm (see [16-19]). In practice, the parameters should be reduced if the value of $L^{(h)}$ remains approximately the same over several iterations or if a fixed number of iterations are performed. The number of iterations to be performed before decreasing the parameters, and the decreasing rate, have to be selected.

3.3. Construction of a feasible schedule

Because of the discrete decision variables and the stopping criterion used in the subgradient method, the solution to the dual problem is generally associated with an infeasible schedule, i.e., some of the capacity constraints might be violated for several time slots. The list-scheduling approach of [19] is used to sonstruct a feasible schedule from this solution. In this list-scheduling procedure, a list is created by ordering operations of all jobs in non decreasing order of their starting times in the solution. Operations are then scheduled on the required machines according to this list as soon as the machines become available. If the capacity constraint for a particular machine is violated at time τ, a greedy heuristic based on the incremental change in J (the original cost function (1)) determines which new operations should begin at that time slot and which ones should be delayed by one time unit. The subsequent operations of these delayed ones are also delayed by one time unit if precedence constraints are violated. The process repeats until a feasible solution is found. The pseudocode for this heuristic can be found in [19].

3.4. Evaluation of the feasible schedule

Once a feasible schedule is obtained, the corresponding value of the objective function J is an upper bound of the optimal objective J^*. The value of the dual function L^*, on the other hand, is a lower bound of J^*. The difference between J^* and L^* is the duality gap. An upper bound of the duality gap is provided by $J - L^*$. The approximate relative duality gap is then $(J - L^*)/L^*$, which is a measure of the suboptimality of the feasible schedule.

4. A dynamic programming algorithm for the relaxed job level subproblems

Let us consider the following one job scheduling problem: A job requires a set of operations to be completed, and each operation can be performed on one machine among several. It is assumed that the processing of operations of the job must verify a set of precedence constraints. The job has a due date and a weight as its unit tardiness penalty. Each machine type has a marginal cost for utilization at each time unit within the time horizon under consideration. The scheduling problem is to determine the machine and the starting time of each operation of the job to minimize the weighted quadratic tardiness of the job and the total cost of using machines to complete the job, where the cost of using machine μ at time τ is $\lambda_{\tau,\mu}$.

Since only one job is considered, for simplicity, the index related to jobs in the notations in the previous section can be dropped. For instance, n denotes the number of operations of the job instead of n_i.

The one job scheduling problem, denoted by **JP**, can then be formulated as follows:

$$\underset{\{m_j\},\{c_j\}}{\text{Minimize}} \left\{ \varphi(c_n) + \sum_{j=1}^{n} \sum_{\tau=c_j-t^j_{m_j}}^{c_j-1} \lambda_{\tau,m_j} \right\} \tag{11}$$

subject to precedence constraints:

$$c_k \leq c_j - t^j_{m_j}, \quad \forall k \in P_j, \quad \forall 1 \leq j \leq n. \tag{12}$$

Note that all relaxed job level subproblems in the Lagrangian relaxation approach of the previous section can be formulated as a one job scheduling problem.

4.1. Optimality conditions

In this subsection, we describe several conditions for optimal solutions of problem JP. They are used for implementing a dynamic programming algorithm for the problem.

For any $\tau = 0, 1, \ldots, H$, for any $1 \leq j \leq n$ and for any $\mu \in \mathcal{M}_j$, let

$$\pi^j_{\tau,\mu} = \begin{cases} \displaystyle\sum_{\theta=\tau-t^j_\mu}^{\tau-1} \lambda_{\theta,\mu}, & \text{if } j < n. \\ \varphi(\tau) + \displaystyle\sum_{\theta=\tau-t^j_\mu}^{\tau-1} \lambda_{\theta,\mu}, & \text{if } j = n. \end{cases}$$

Therefore, $\pi^j_{\tau,\mu}$ is the cost of completing operation j at time τ on machine μ. With these new notations, the objective function of Problem JP can be rewritten as

$$\underset{\{m_j\},\{c_j\}}{\text{Minimize}} \sum_{j=1}^{n} \pi^j_{c_j,m_j} \tag{13}$$

We now show that the problem can be solved using dynamic programming technique. For each operation j, let $\mathcal{P}_j$ denote the set containing its (not necessarily immediate) precedent operations and itself.

Let us define a problem $\mathbf{JP_k(x,u)}$, with parameters $1 \leq k \leq n$, $0 \leq x \leq H$ and $u \in \mathcal{M}_k$ as follows.

$$\underset{\{m_j\},\{c_j\}}{\text{Minimize}} \sum_{j \in \mathcal{P}_k} \pi^j_{c_j,m_j} \tag{14}$$

subject to (12) and $c_k = x$, $m_k = u$.

Let $f_k(x,u)$ denote the optimal criterion value of Problem $\mathbf{JP_k(x,u)}$. $f_k(x,u)$ is the minimal cost to complete the prodecessors of operation k and itself such that operation k is completed *exactly* at time x on machine u. It is possible that no solution exists for problem $\mathbf{JP_k(x,u)}$. In this case, $f_k(x,u)$ is set to $+\infty$, without loss of generality.

According to the precedence constraints, if $c_k = x$ and $m_k = u$, we should have $c_j \leq x - t^k_u$, for any $j \in P_k$. Considering further more the constraints $m_j \in \mathcal{M}_j$ and the fact that $\mathcal{P}_k = \{k\} \bigcup \bigcup_{j \in P_k} \mathcal{P}_j$, this remark leads to the following formulae

$$f_k(x,u) = \pi^k_{x,u}; \quad \forall k \text{ such that } P_k = \emptyset; \tag{15}$$

$$f_k(x,u) = \pi^k_{x,u} + \sum_{j \in P_k} \min_{v \in \mathcal{M}_j} g_j(x - t^k_u, v). \quad \forall k \text{ such that } P_k \neq \emptyset. \tag{16}$$

where,

$$g_j(y,v) = \min_{0 \leq z \leq y} f_j(z,v). \tag{17}$$

It can be seen that (15) and (16) provide respectively the boundary condition and the recurrence formula for the dynamic programming approach. The physical interpretation of the function $g_k(x,u)$ is the minimal cost to complete the predecessors of operation k and itself such that operation k is completed *at latest* at time x on machine u.

As a result, the optimal function value of the one job problem is given by $\min_{\mu \in \mathcal{M}_n} g_n(H,\mu)$. The corresponding optimal solution can be obtained by a backward method.

4.2. Computational complexity of the dynamic programming algorithm

For each triplet (k,x,u), in order to compute $f_k(x,u)$, we should consider all the immediate predecessors of operation k. For each operation $j \in P_k$, the number of v values is $|\mathcal{M}_j|$. In addition, for each value of k, the number of x's is $O(H)$ and the number of u's is $|\mathcal{M}_k|$. Therefore, the overall complexity of the algorithm is $O(H \sum_{k=1}^{n} |\mathcal{M}_k| \sum_{j \in P_k} |\mathcal{M}_j|)$. According to (17), we have $g_k(x,u) = \min(g_k(x-1,u), f_k(x,u))$. Therefore, functions $g_k(x,u)$ can be computed at the same time as functions $f_k(x,u)$. Their computation needs only a comparison and does not increase the order of complexity.

5. "Min-Max" Criteria

As we pointed out in the introduction, the application of Lagrangian relaxation approach has been limited to "min-sum" type criteria, since with "min-max" type criteria, the problem cannot be decomposed into job level subproblems. In this section, we show that with our dynamic programming approach, we can solve scheduling problems with some "min-max" criteria. We show that the minimization of makespan and the minimization of maximal lateness can be performed by Lagrangian relaxation methods. Both these criteria are of "Min-Max" type. Indeed, with these criteria, the relaxed problem cannot be completely decomposed. However, we can construct job

level subproblems as for "min-sum" criteria. Instead of optimizing the job level subproblem, we obtain a set of solutions, each of with is an optimal solution for a given parameter. The global optimal solution of the relaxed problem is obtained by choosing the values of these parameters. We show that they can be chosen with a pseudopolynomial algorithm.

More precisely, for each job i ($1 \leq i \leq N$), we can solve a job level problem by replacing n (resp. t_μ^j, $\mathcal{M}_j$, ...) in Section 4 with n_i (resp. $t_\mu^{i,j}$, $\mathcal{M}_{i,j}$, ...), and by replacing the function φ by 0. We then obtain a function $g_{i,n_i}(x,u)$ with parameters x and u, for each $0 \leq x \leq H$ and for each $u \in \mathcal{M}_{i,n_i}$. The function $g_{i,n_i}(x,u)$ gives the cost of completing job i such that operation (i,n_i) is completed at latest at time x on machine u.

We consider now respectively the minimization of makespan and the minimization of maximal lateness.

When the criteria to minimize is the makespan, the objective function of the relaxed problem becomes, instead of (5):

$$\underset{\{m_{i,j}\},\{c_{i,j}\}}{\text{Minimize}} \left\{ \left[\max_{1\leq i\leq N} C_i + \sum_{i=1}^{N} \sum_{j=1}^{n_i} \sum_{\tau=c_{i,j}-t^{i,j}_{m_{i,j}}}^{c_{i,j}-1} \lambda_{\tau,m_{i,j}} \right] - \sum_{\mu=1}^{M} \sum_{\tau=0}^{H} \lambda_{\tau,\mu} \right\} \tag{18}$$

The term outside the brackets does not depend on decision variables. We then have to optimize only the term between the brackets. Let V^* be the optimal value of this term. We then have the following theorem.

Theorem 1. *There is a $0 \leq x \leq H$ such that $V^* = x + \sum_{i=1}^{N} \min_{u\in\mathcal{M}_{i,n_i}} g_{i,n_i}(x,u)$.*

Proof. Consider an optimal solution $\{c^*_{i,j}, m^*_{i,j} | 1 \leq j \leq n_i,\ 1 \leq i \leq N\}$. According to the assumption that H is large enough, we have $0 \leq C_i^* = c^*_{i,n_i} \leq H$, $\forall 1 \leq i \leq N$. By the optimality of the solution, we have

$$\sum_{j=1}^{n_i} \sum_{\tau=c^*_{i,j}-t^{i,j}_{m^*_{i,j}}}^{c^*_{i,j}-1} \lambda_{\tau,m^*_{i,j}} = f_{i,n_i}(C_i^*, m^*_{i,n_i}). \quad \forall 1 \leq i \leq N.$$

Let $x = \max_{1\leq i\leq N} C_i^*$. We now prove that

$$f_{i,n_i}(C_i^*, m^*_{i,n_i}) = \min_{\mu\in\mathcal{M}_{i,n_i}} g_{i,n_i}(x,\mu). \quad \forall 1 \leq i \leq N.$$

For any job $1 \leq i \leq N$, from equation (17), we have $g_{i,n_i}(x,u) \leq f_{i,n_i}(C_i^*, u)$, $\forall u \in \mathcal{M}_{i,n_i}$, which leads to $\min_{u\in\mathcal{M}_{i,n_i}} g_{i,n_i}(x,u) \leq f_{i,n_i}(C_i^*, m^*_{i,n_i})$. If $\min_{u\in\mathcal{M}_{i,n_i}} g_{i,n_i}(x,u) \neq f_{i,n_i}(C_i^*, m^*_{i,n_i})$, there would be a douplet $(y,u) \neq (C_i^*, m^*_{i,n_i})$ such that $0 \leq y \leq x$, $u \in \mathcal{M}_{i,n_i}$ and $f_{i,n_i}(y,u) < f_{i,n_i}(C_i^*, m^*_{i,n_i})$, which is in contradiction with the optimality of the solution. □

According to this theorem, the optimal criterion value of the relaxed problem is given by

$$\min_{0\leq x\leq H} \left\{ x + \sum_{i=1}^{N} \min_{u\in\mathcal{M}_{i,n_i}} g_{i,n_i}(x,u) \right\}.$$

As soon as the functions $g_{i,n_i}(x,u)$ is known for each job i, this problem can be solved by enumrating all the possible values for x and u values for each given x. The complexity is then $O(H\sum_{i=1}^{N}|\mathcal{M}_{i,n_i}|)$.

Similarly, if the criterion considered is to minimize maximal lateness, the relaxed problem can also be solved. By considering that $g_{i,n_i}(x,u)=+\infty$ for any $u\in\mathcal{M}_{i,n_i}$ and for any x such that $x<0$ or $x>H$, the optimal criterion value is given by

$$\min_{-D_{\max}\le x\le H-D_{\min}}\left\{x+\sum_{i=1}^{N}\min_{u\in\mathcal{M}_{i,n_i}}g_{i,n_i}(x+D_i,u)\right\}.$$

where D_i is the due date of job i and $D_{\max}=\max_{1\le i\le N}D_i$, $D_{\min}=\min_{1\le i\le N}D_i$.

6. Test results

6.1. Example 1

The first example is example 1 in [19]. It is a job shop scheduling problem with 4 equally weighted jobs and 3 different machines. The total number of operations is 24. The scheduling horizon is 25 days. All machines are available on day 1 and all jobs are due on day 0. In [19], it takes 12.7 CPU seconds on an IBM 3090 mainframe computer to obtain a lower bound 469.7 and a feasible solution with cost J=475.0. The relative duality gap is then 1.13%.

The problem has been solved using the approach presented in this paper. All the multipliers are initialized at zero and the initial upper bound is set to 600. It needs 99 iterations, and only 0.46 seconds on IBM 6091/19 microcomputer, to obtain a very tight lower bound 474.9998, and a feasible solution with cost J=475.0. The duality gap is then 0.000042%. In fact, the feasible solution is actually optimal, taking into account the fact that cost is integer.

6.2. Example 2

The second example is a randomly generated 10 job 5 machine (denoted as 10×5 as in the literature) job shop scheduling problem. The jth operation of job i is processed on machine $(i+j)(\mathrm{mod}\ 5)$. The processing time of each operation and the due date of each job are randomly generated from the uniform distribution [1,10] and the uniform distributuion [1,25] respectively. These data are shown in Table 1. The weights of all jobs are 1 and the scheduling horizon is 100. All the multipliers are initialized at zero as described in Example 1 and the initial upper bound is 15 000. It takes 220 iterations and 13.58 CPU seconds on an IBM 6091/19 microcomputer, to obtain a lower bound 9 343.1914 and a feasible solution with cost J=10 250. The relative duality gap is then 9.7%.

6.3. Example 3

The third example is a randomly generated 20×10 job shop scheduling problems. The jth operation of job i is processed on machine $(i+j)(\mathrm{mod}\ 10)$. The processing time of each operation and the due date of each job are randomly generated from the uniform distribution [1,10] and the uniform distributuion [1,100] respectively. These data are shown in Table 2. The weights of all jobs are 1 and the scheduling horizon is 200. All the multipliers are initialized at zero as described in Example 1 and the initial upper bound is 60 000. It takes 227 iterations and 99 CPU seconds on an IBM 6091/19 microcomputer, to obtain a lower bound 35 625.2266 a feasible solution with cost J=42 950. The relative duality gap is 20.56%.

Table 1: Data of Example 2

	Operation					
Job	1	2	3	4	5	Due Date
1	4	8	4	5	3	9
2	8	1	6	2	4	24
3	7	1	9	2	2	10
4	5	6	6	3	4	24
5	3	7	3	1	4	1
6	8	9	6	8	1	13
7	5	5	8	3	9	7
8	3	6	6	2	8	1
9	3	2	4	4	6	15
10	7	6	5	5	6	6

Table 2: Data of Example 3

	Operation										
Job	1	2	3	4	5	6	7	8	9	10	Due Date
1	4	8	4	5	3	8	1	6	2	4	70
2	7	1	9	2	2	5	6	6	3	4	35
3	3	7	3	1	4	8	9	6	8	1	90
4	5	5	8	3	9	3	6	6	2	8	96
5	3	2	4	4	6	7	6	5	5	6	13
6	3	9	4	9	1	5	3	1	6	2	99
7	8	1	3	3	9	5	7	3	8	7	6
8	5	9	7	4	3	4	3	7	7	6	62
9	8	4	8	8	2	9	4	5	1	9	4
10	8	7	4	7	7	4	2	5	3	8	38
11	5	1	8	1	8	3	3	9	9	8	53
12	5	2	6	3	3	3	2	1	1	3	29
13	4	7	2	5	1	4	5	2	2	9	57
14	4	2	2	6	1	8	1	4	2	9	61
15	9	5	8	7	9	5	4	9	7	8	82
16	7	6	7	6	3	9	6	1	4	6	45
17	2	4	6	1	8	1	7	3	9	4	3
18	3	2	6	3	3	3	9	2	3	5	48
19	3	8	3	7	8	8	6	1	9	3	29
20	6	8	8	4	7	1	8	6	8	4	30

Remark. The test results on Examples 2 and 3 and other randomly generated job shop scheduling problems show that the relative duality gap obtained by using Lagrangian relaxation increases as the machine number increases and the gap is larger than 10% for some of the problems. It is not clear now that the larger duality gap is caused by ineffectiveness of the list-scheduling approach for constructing a feasible schedule or is an inherent quality of the Lagrangian relaxation approach. However, since the job shop scheduling problems are extremely difficult to be solved, a relative duality gap of the value, such as 20% , can still be thought as within a reasonable limit.

7. Conclusion

In this paper, we propose an efficient pseudopolynomial time dynamic programming algorithm to relaxed job level subproblems, and then show that in extending the Lagrangian relaxation technique to job shop scheduling problems, the relaxation of the precedence constraints is unnecessary. This algorithm results in a much more efficient Lagrangian relaxation approach to job-shop scheduling problems, in which the solution oscillation due to decomposing each relaxed job level problem into a set of operation level subproblems is completely avoided and great computation time savings are carried out. The new Lagrangian relaxation approach has a significant improvement over existing Lagrangian-relaxation-based job-shop scheduling approaches and is more practical for implementation in actual manufacturing environments.

The test results on randomly generated job shop scheduling problems show that the relative duality gap obtained by using Lagrangian relaxation increases as the machine number increases and the gap is larger than 10% for some of the problems. The larger relative duality gap is likely to be caused by ineffectiveness of the list-scheduling approach for constructing a feasible schedule for large-sized job shop problems with complicated machine routes of jobs. So, one subject for further investigation is to develop a more efficient approach for constructing a feasible solution from the solution of the dual problem.

In addition, although the precedence constraints considered in this paper are general enough for job shop scheduling problems in most manufacturing/assembling systems, we'd like to find an efficient algorithm for the relaxed job level subproblems under generic precedence constraints. This is one subject of our current investigation.

References

[1] E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan and D. B. Shmoys, "Sequencing and scheduling: Algorithm and complexity", in: *Handbooks in Operations Research and Management Science*, Vol. **4**. North-Holland, 1991.

[2] S. C. Graves, "A review of production scheduling", *Operations Research*, Vol. **18** (1981), pp. 841-852, 1981.

[3] M. Held and R. M. Karp, "A dynamic programming approach to sequencing problems", *J. Soc. Indust. Appl. Math.*, Vol. **10** (1962), pp. 196-210.

[4] C. N. Potts, L. N. Van Wassenhove, "Dynamic programming and decomposition approaches for the single machine totoal tardiness problem", *European Journal of Operational Research*, Vol. **32**, 1987, pp.405-414.

[5] N. G. Hall and M. E. Posner, "Earliness-tardiness scheduling problems, I: Weighted deviation of completion times about a common due date", *Operations Research*, Vol. **39** (1991), pp. 836-846.

[6] C. N. Potts, L. N. Van Wassenhove, "A branch and bound algorithm for the total weighted tardiness problem", *Operation Research*, Vol. **33** (1985), pp. 363-377.

[7] M. L. Fisher, "Optimal solution of scheduling problems using Lagrange multipliers, Part I", *Operation Research*, Vol. **21** (1973), pp. 1114-1127.

[8] M. L. Fisher, "A dual algorithm for the one-machine scheduling problem", *Mathematical programming*, Vol. **11** (1976), pp. 229-251.

[9] B.J. Lageweg, J.K. Lenstra, A.H.G. Rinnooy Kan, "A general bounding scheme for the permutation flow-shop problem", *Operations Research*, Vol. **26** (1978), pp. 53-67.

[10] B.J. Lageweg, J.K. Lenstra, A.H.G. Rinnooy Kan, "Job shop scheduling by implicit enumeration", *Management Science*, Vol. **24** (1977), pp. 441-450.

[11] J. Carlier and E. Pinson, "An algorithm for solving the job-shop problem", *Manag. Sci.*, Vol. **35** (1989), pp. 164-176.

[12] H.G. Campbell, R.A. Dudek and M. L. Smith, "A heuristic algorithm for the n job, m machine sequencing problem", *Management Sciences*, Vol. **16** (1970), pp. 630-637.

[13] J. Adams, E. Balas and D. Zawack, "The shifting bottleneck procedure for job shop scheduling", *Management Science*, Vol. **34** (1988), pp. 391-401.

[14] I. H. Osman, C. N. Potts, "Simulated annealing for permutation flow-shop scheduling", *OMEGA Int . J. of Management Science*, Vol. **17** (1989), pp. 551-557.

[15] P. J. M. Van Laarhoven, E. H. L. Aarts, J. K. Lenstra, "Job shop scheduling by simulated annealing", Report OS-R8809, Center for Mathematics and Computer Sci., Amsterdam, 1988.

[16] P. B. Luh, D.J. Hoitomt, E. Max, and K.P. Pattipati, "Scheduling generation and reconfiguration for parallel machines", *IEEE Trans. Robotics and Automat.*, Vol. **6** (1990), pp. 687-696.

[17] P.B. Luh and D. Hoitomt, "Scheduling of manufacturing systems using the Lagrangian relaxation techniques", *IFAC' Workshop on Discrete Event System Theory and Applications*, Shenyang, China, 1991.

[18] D. J. Hoitomt, P. B. Luh, K.R. Pattipati, "Job shop scheduling", *Proceedings of the First International Conference on Automation Technology*, Taipei, Taiwan, pp. 565-574, 1990.

[19] D.J. Hoitomt, P. B. Luh, K.R. Pattipati, "A practical approach to job-shop scheduling problems", *IEEE Trans. on Robotics and Automation*, Vol. **9**(1993), pp. 1-13.

[20] C. S. Czerwinski and P.B. Luh, "An improved Lagrangian relaxation technique for job shop scheduling", *Proceedings of the IEEE Conference on Decision and Control*, Vol. **1** (1992), pp. 771-776.

[21] Chen Haoxun, J.M. Proth, Hu Baosheng. "A scheduling framework based on Petri net modeling and sequential Lagrangian relaxation approach", *Proceeding of the Rensselaer's Fourth International Conference on Computer Integrated Manufacturing and Automation Technology*, Resselaer Polytechnic Institute, New York, Oct. 10-12, 1994.

[22] R. N. Tomastik, P.B. Luh, "The facet ascending algorithm for integer programming problems", *Proceedings of the 32nd Conference on Decision and Control*, San Antonio, Texas, December, 1993, pp. 2880-2884.

[23] B. T. Polyak, "Minimization of unsmooth functionals", *USSR Comput. Math. Phys.*, Vol. **9** (1969), pp. 14-29.

[24] A. M. Geoffrion, "Lagrangian relaxation for integer programming", *Math. Programm. Study*, Vol. **2** (1974), pp. 82-114.

Unité de recherche INRIA Lorraine
Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - B.P. 101 - 54602 Villers lès Nancy Cedex (France)

Unité de recherche INRIA Rennes - IRISA, Campus universitaire de Beaulieu 35042 Rennes Cedex (France)
Unité de recherche INRIA Rhône-Alpes - 46, avenue Félix Viallet - 38031 Grenoble Cedex 1 (France)
Unité de recherche INRIA Rocquencourt - Domaine de Voluceau - Rocquencourt - B.P. 105 - 78153 Le Chesnay Cedex (France)
Unité de recherche INRIA Sophia Antipolis - 2004, route des Lucioles - B.P. 93 - 06902 Sophia Antipolis Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt - B.P. 105 - 78153 Le Chesnay Cedex (France)

ISSN 0249 - 6399