JOURNAL OF OPTIMIZATION THEORY AND APPLICATIONS: Vol. 100, No. 3, pp. 699-712, MARCH 1999

Surrogate Gradient Algorithm for Lagrangian Relaxation[1,2]

X. ZHAO,[3] P. B. LUH,[4] and J. WANG[5]

Communicated by W. B. Gong and D. D. Yao

Abstract. The subgradient method is used frequently to optimize dual functions in Lagrangian relaxation for separable integer programming problems. In the method, all subproblems must be solved optimally to obtain a subgradient direction. In this paper, the surrogate subgradient method is developed, where a proper direction can be obtained without solving optimally all the subproblems. In fact, only an approximate optimization of one subproblem is needed to get a proper surrogate subgradient direction, and the directions are smooth for problems of large size. The convergence of the algorithm is proved. Compared with methods that take effort to find better directions, this method can obtain good directions with much less effort and provides a new approach that is especially powerful for problems of very large size.

Key Words. Lagrangian relaxation, integer programming, surrogate subgradient, job shop scheduling.

1. Introduction

Playing a fundamental role in constrained optimization in economics and mathematics over the decades, Lagrangian relaxation is particularly powerful for the optimization of separable nonlinear programming problems or integer programming problems. In a nutshell, the key idea of the approach

[1]This paper is dedicated to Professor Yu-Chi Ho for his 65th birthday.

[2]This work was supported in part by National Science Foundation Grant DMI-9500037 and the Cannondale Corporation.

[3]Graduate Student, Department of Electrical and System Engineering, University of Connecticut, Storrs, Connecticut.

[4]Professor, Department of Electrical and System Engineering, University of Connecticut, Storrs, Connecticut.

[5]Graduate Student, Department of Electrical and System Engineering, University of Connecticut, Storrs, Connecticut.

0022-3239/99/0300-0699$16.00/0

is decomposition and coordination, where decomposition is based on the separability of models, and coordination based on the pricing concept of a market economy. In this method, coupling constraints are first relaxed through the introduction of Lagrange multipliers, and the relaxed problem can be decomposed into many smaller subproblems. Given the set of multipliers, these subproblems are easier than the original problem, and can be solved efficiently. Multipliers are then adjusted iteratively based on the levels of constraint violation. The resolution of the original problem is thus done through a two-level iterative approach, where the low level consists of solving individual subproblems. Coordination of the subproblem solutions is performed through the updating of the Lagrange multipliers at the high level. In the optimization terminology, the concave dual function is maximized iteratively. In this process, the subproblem solutions will tend to an optimal feasible solution, while the dual function itself provides a lower bound to the optimal primal cost (Ref. 1).

When applying Lagrangian relaxation to integer programming, techniques such as the subgradient, bundle, and cutting plane methods are used often to maximize the dual function, since the dual function is polyhedral concave and nondifferentiable. The subgradient method is the most widely used method, where the subgradient direction is obtained after all the subproblems are solved and the multipliers are updated along this subgradient direction. The bundle method can provide better directions than the subgradient method. However, to obtain each direction, it may require solving all the subproblems many times (Ref. 2). For problems of large size, the solution of the subproblems can be complicated and takes the majority of the computation time. For example, it has been reported that more than 70% of the total CPU time is spent on solving subproblems for job shop scheduling problems (Ref. 3). Therefore, it is desirable to obtain a good direction with less effort than solving all the subproblems many times to obtain an optimized direction.

Based on this idea, the interleaved subgradient method solves only one subproblem per iteration to obtain a direction and then updates the multipliers (Ref. 4). Numerical results show that the method converges faster than the subgradient method, although the algorithm convergence was not established.

In this paper, the surrogate subgradient method is developed, and a proper surrogate subgradient direction to update the multipliers can be obtained without solving all the subproblems. In fact, only an approximate solution of one subproblem is needed to obtain a surrogate subgradient direction. The convergence of the algorithm is proved in Section 4 for separable nonlinear programming or integer programming problems. This method includes the interleaved subgradient method as a special case and provides

a framework allowing creative variations. Compared with methods such as the conjugate method (Ref. 5) or the bundle method, that take effort to find a better direction, this method saves much effort in obtaining a direction. It is also shown that, for large problems, the surrogate gradient directions are smooth from one iteration to the next; therefore, this method can avoid the notorious zigzagging difficulties associated with the subgradient method and leads to fast convergence. The modified surrogate gradient method is developed in Section 5, and it can provide better directions than the surrogate gradient method for problem of small size. Testing results on a simple nonlinear programming problem and several realistic job shop scheduling problems, provided in Section 6, show that the surrogate methods present significant improvements over the frequently used subgradient method.

2. Problem Description and Formulation

2.1. Integer Programming Problem. The separable integer programming problem under consideration can be described as follows:

$$(\text{IP}) \quad \min_{x^l \le x \le x^u} \quad J_{\text{IP}} \equiv \sum_{i=1}^{I} J_i(x_i), \tag{1}$$

$$\text{s.t.} \quad Ax \le b, \tag{2a}$$

$$x_i \in Z^{n_i}, \qquad i = 1, \ldots, I. \tag{2b}$$

Here, $x = [x_1, x_2, \ldots, x_I]^T$ is an $n \times 1$ decision variable with $n = \sum_{i=1}^{I} n_i$; x^l and x^u are the lower and upper bounds of x; and Z is the set of integers. The $m \times n$ matrix A is of the form $[a_1, a_2, \ldots, a_I]$, where a_i is an $m \times n_i$ matrix, b is an $m \times 1$ vector, and the objectives $\{J_i(x_i)\}$ are possibly nonlinear functions.

2.2. Lagrangian Relaxation. The m constraints $Ax \le b$ couple the decision variables x_i, thus making (IP) difficult to solve. The Lagrangian relaxation of (IP) is given by

$$L(\lambda) \equiv \min_{x^l \le x \le x^u,\, x \in Z^n} \left[\sum_{i=1}^{I} J_i(x_i) + \lambda^T (Ax - b) \right]. \tag{3}$$

Here, λ is an $m \times l$ vector of Lagrangian multipliers and the function $L(\lambda)$ is the Lagrangian dual. Since the decision variables are decoupled through the introduction of the multipliers λ, (3) can be written in terms of individual

subproblems as follows:

$$L_i(\lambda) = \min_{x_i^l \le x_i \le x_i^u, x_i \in Z^{n_i}} [J_i(x_i) + \lambda^T (a_i x_i)], \tag{4}$$

$$L(\lambda) = \sum_{i=1}^{I} L_i(\lambda) - b^T \lambda. \tag{5}$$

The minimization in (3) is easier than in (IP), since each subproblem can be solved independently.

The Lagrangian dual problem is (Ref. 6)

$$(\text{LD}) \quad \max_{\lambda \ge 0} L(\lambda), \tag{6}$$

and the optimal solution is denoted as $L^* = L(\lambda^*)$.

3. Subgradient and Surrogate Subgradient

Since problem (LD) in (6) is polyhedron concave and nondifferentiable, the subgradient method is commonly used to maximize the dual function. In the subgradient method, the multipliers are updated by

$$\lambda^{k+1} = \lambda^k + s^k g^k. \tag{7}$$

Here, $g^k = g(\lambda^k)$ is the subgradient of $L(\lambda)$ at λ^k and is given by

$$g(\lambda^k) = Ax(\lambda^k) - b = \sum_{i=1}^{I} a_i x_i(\lambda^k) - b, \tag{8}$$

where

$$x_i(\lambda^k) = \arg \min_{x_i^l \le x_i \le x_i^u, x_i \in Z^{n_i}} [J_i(x_i) + (\lambda^k)^T (a_i x_i)]. \tag{9}$$

The stepsize s^k satisfies

$$0 < s^k < 2(L^* - L^k)/\|g^k\|^2. \tag{10}$$

Although the subgradient method is not an ascending algorithm, the subgradient satisfies (Ref. 7)

$$0 \le L^* - L^k \le (\lambda^* - \lambda^k)^T g(\lambda^k). \tag{11}$$

Thus, the subgradient direction is in acute angle with the direction toward λ^*, and the distance between the current multipliers and the optimal λ^* can be decreased step by step.

According to (8) and (9), the subgradient method requires the solution of all the subproblems to obtain a search direction, and this can be very time consuming especially for problems of large size. Therefore, it is desirable

to obtain a proper direction with less effort. The main idea of the surrogate subgradient method is to obtain a proper surrogate subgradient direction without solving all the subproblems.

As an extension of the dual in (3), the surrogate dual is introduced,[6]

$$\tilde{L}(\lambda, x) \equiv \sum_{i=1}^{I} J_i(x_i) + \lambda^T(Ax - b), \qquad \text{with } x^l \le x \le x^u, \qquad x \in Z^n. \tag{12}$$

Compared with the dual, the surrogate dual does not require the solution of all subproblems. The corresponding surrogate subgradient is defined as

$$\tilde{g}(x) \equiv Ax - b = \sum_{i=1}^{I} a_i x_i - b. \tag{13}$$

It will be shown in the following proposition that, when the surrogate dual is less than the optimal dual L^*, the surrogate subgradient is in acute angle with the direction toward λ^*; therefore it is a proper direction.

Proposition 3.1. Given the current point (λ^k, x^k), if the surrogate dual is less than the optimal dual, i.e.,

$$\tilde{L}^k = \tilde{L}(\lambda^k, x^k) < L^*, \tag{14}$$

then the surrogate subgradient satisfies

$$0 \le L^* - \tilde{L}^k \le (\lambda^* - \lambda^k)^T \tilde{g}(x^k). \tag{15}$$

Proof. From the definition of the surrogate dual in (12),

$$\tilde{L}(\lambda, x^k) = \sum_{i=1}^{I} J_i(x_i^k) + \lambda^T(Ax^k - b) = \tilde{L}^k + (\lambda - \lambda^k)^T(Ax^k - b). \tag{16}$$

Since minimization is performed in deriving $L(\lambda)$ in (3), the surrogate dual is always greater than or equal to the dual,

$$L(\lambda) \le \tilde{L}(\lambda, x). \tag{17}$$

The above is also true at (λ^*, x^k), i.e.,

$$L^* = L(\lambda^*) \le \tilde{L}(\lambda^*, x^k). \tag{18}$$

From (16), this can be written as

$$L^* \le \tilde{L}^k + (\lambda^* - \lambda^k)^T(Ax^k - b). \tag{19}$$

[6]This is different from the surrogate dual used in branch-and-bound methods (Ref. 8).

Given (14) and (13), this yields

$$0 \leq L^* - \tilde{L}^k \leq (\lambda^* - \lambda^k)^T \tilde{g}(x^k),$$

and (15) is proved. □

4. Surrogate Subgradient Method

Based on Proposition 3.1, if an appropriate optimization is performed so that (14) is satisfied from one iteration to the next, then (15) is guaranteed. This implies that the algorithm can find a proper direction, and the distance between the current multipliers and the optimal λ^* can be decreased step by step. This idea leads to the following surrogate subgradient method where only an approximate optimization is required for subproblems to obtain a surrogate subgradient. The basic steps in the surrogate subgradient method are described below.

Step 0. Initialize. Assume λ^0 and solve the subproblems to obtain x^0, i.e.,

$$x^0 = \arg \min_{x^l \leq x \leq x^u, x \in Z^n} \left\{ \sum_{i=1}^{I} J_i(x_i) + (\lambda^0)^T \left(\sum_{i=1}^{I} a_i x_i - b \right) \right\}. \tag{20}$$

Thus, the surrogate dual is set to the dual and the surrogate subgradient is the subgradient, i.e.,

$$\tilde{L}(\lambda^0, x^0) = L(\lambda^0), \tag{21}$$

$$\tilde{g}(x^0) = g(\lambda^0). \tag{22}$$

Step 1. Update the Multipliers. Given the current point (λ^k, x^k) at the kth iteration, the surrogate dual is

$$\tilde{L}^k = \tilde{L}(\lambda^k, x^k) = \sum_{i=1}^{I} J_i(x_i^k) + (\lambda^k)^T (Ax^k - b), \tag{23}$$

with

$$x^l \leq x^k \leq x^u, \qquad x^k \in Z^n. \tag{24}$$

The Lagrangian multipliers are updated according to

$$\lambda^{k+1} = \lambda^k + s^k \tilde{g}^k, \tag{25}$$

where $\tilde{g}^k$ is the surrogate subgradient given by

$$\tilde{g}^k = \tilde{g}(x^k) = Ax^k - b, \tag{26}$$

with stepsize s^k satisfying

$$0 < s^k < (L^* - \tilde{L}^k)/\|\tilde{g}^k\|^2. \tag{27}$$

Step 2. Perform an Approximate Optimization. Given λ^{k+1}, perform an approximate optimization to obtain x^{k+1} such that x^{k+1} satisfies

$$\tilde{L}(\lambda^{k+1}, x^{k+1}) < \tilde{L}(\lambda^{k+1}, x^k) = \sum_{i=1}^{I} J_i(x_i^k) + (\lambda^{k+1})^T(Ax^k - b). \tag{28}$$

If such an x^{k+1} cannot be obtained, set $x^{k+1} = x^k$.

Step 3. Check the Stopping Criteria. If the criteria given by

$$\|\lambda^{k+1} - \lambda^k\| < \epsilon_1, \tag{29}$$

$$\|x^{k+1} - x^k\| < \epsilon_2, \tag{30}$$

are met, then stop; otherwise, go to Step 1. Stopping criteria can also be based on CPU time, number of iterations, and so on.

It will be shown next that (14) is always satisfied for the surrogate subgradient method.

Proposition 4.1. For the surrogate subgradient method, the surrogate dual is always less than the optimal dual, i.e.,

$$\tilde{L}^k < L^*, \qquad \text{for all } k. \tag{31}$$

Proof. This proposition is proved by induction. For iteration 0, the surrogate dual is initialized to be the dual, and (14) holds. For iteration k, if (14) holds, then (28) and (23) yield

$$\tilde{L}^{k+1} < \tilde{L}^k + (\lambda^{k+1} - \lambda^k)^T(Ax^k - b). \tag{32}$$

According to (25) and (26), the right-hand side of (32) can be rewritten as

$$\tilde{L}^k + (\lambda^{k+1} - \lambda^k)^T(Ax^k - b) = \tilde{L}^k + s^k\|\tilde{g}^k\|^2. \tag{33}$$

Considering the range of stepsize in (27), the right side of (33) satisfies

$$\tilde{L}^k + s^k\|\tilde{g}^k\|^2 < L^*. \tag{34}$$

Thus, one can see that (14) holds at iteration $k+1$, and (31) is proved. □

Based on Proposition 3.1 and 4.1, the surrogate subgradient is always a proper direction. Therefore, we have the following theorem.

Theorem 4.1. In the surrogate subgradient method, the multipliers move closer to λ^* step by step, i.e.,

$$\|\lambda^* - \lambda^{k+1}\| < \|\lambda^* - \lambda^k\|, \qquad \text{for all } k. \tag{35}$$

Proof. From (25),

$$\|\lambda^* - \lambda^{k+1}\|^2 = \|\lambda^* - \lambda^k\|^2 - 2s^k(\lambda^* - \lambda^k)^T\tilde{g}^k + (s^k)^2\|\tilde{g}^k\|^2. \tag{36}$$

From (15), this yields

$$\|\lambda^* - \lambda^{k+1}\|^2 \leq \|\lambda^* - \lambda^k\|^2 - 2s^k(L^* - \tilde{L}^k)^T + (s^k)^2\|\tilde{g}^k\|^2, \tag{37}$$

which can be written as

$$\|\lambda^* - \lambda^{k+1}\|^2 \leq \|\lambda^* - \lambda^k\|^2 - s^k[2(L^* - \tilde{L}^k)^T - s^k\|\tilde{g}^k\|^2]. \tag{38}$$

For the range of stepsizes in (27), the bracket term in (38) is greater than zero. Thus, (35) is proved. □

It can be easily shown that, if $\lambda^k = \lambda^{k+1}$ and $x^k = x^{k+1}$, then

$$L(\lambda^k) = L^*,$$

and x^k is the minimum solution of the subproblems given λ^k. Thus, (λ^k, x^k) is the optimal solution of the dual problem.

The major difference between the surrogate subgradient method and the subgradient method is that all subproblems are solved in the subgradient method. In the surrogate subgradient method, however, only an approximate optimization of the subproblems is needed according to (28). Since there are many ways to implement an approximate optimization, this method provides a framework allowing creative variations. In fact, the interleaved subgradient method that solves one subproblem at a time satisfies (28) and is a special case of the surrogate subgradient method. Thus, we have the following theorem.

Theorem 4.2. The interleaved subgradient method is a special case of the surrogate subgradient method.

Compared with solving all the subproblems, the computational requirements for the approximate optimization to satisfy (28) are much smaller. For example, if there are I subproblems, the effort to obtain a direction for the interleaved subgradient method is only $1/I$ of that required for the subgradient method. The effort will be even less than $1/I$ if an approximate optimization is performed for only one subproblem at a time. This is a major computational saving for problems with many complicated subproblems.

The effort to obtain a direction is one thing; the quality of the direction obtained is another thing. Although it is difficult to quantify the quality of the surrogate subgradient directions, it can be shown that the surrogate subgradient directions tend to be smooth when there are many subproblems. Since only one subproblem is solved and all the other solutions are kept the same, the new direction is changed only partially as compared to the previous direction. Thus, the surrogate subgradient directions are smooth from one iteration to the next when I is large, and can avoid the notorious zigzagging difficulties associated with the subgradient method. For small problem, the surrogate directions can be improved by using the modified surrogate subgradient method to be developed next.

5. Modified Surrogate Subgradient Method

The modified gradient method was developed to improve the directions of the gradient method. In the modified gradient method, a direction is a linear combination of the current gradient and the direction used at the previous iteration (Ref. 9). It is proved that the modified direction forms a smaller (or equal) angle with the direction toward the optimal λ^* than does the gradient direction. Based on this idea, the modified surrogate subgradient method is developed to improve the directions of the surrogate subgradient method. It is a variation of the surrogate subgradient method with Step 1 changed to the following Step 1′.

Step 1′. Update Multipliers. In the modified surrogate subgradient method, the multipliers are updated by

$$\lambda^{k+1} = \lambda^k + s^k \tilde{d}^k, \tag{39}$$

where s^k is the stepsize and $\tilde{d}^k$ the modified surrogate subgradient. The stepsize satisfies

$$0 < s^k \leq (L^* - \tilde{L}^k)/\|\tilde{d}^k\|^2. \tag{40}$$

The modified surrogate subgradient is given by

$$\tilde{d}^0 = \tilde{g}^0, \tag{41}$$

$$\tilde{d}^k = \tilde{g}^k + \gamma^k \tilde{d}^{k-1}, \qquad k = 1, 2, \ldots. \tag{42}$$

In (42), $\tilde{g}^k$ is the current surrogate subgradient, γ is given by

$$\gamma^k = \max[0, -\beta((\tilde{d}^{k-1})^T \tilde{g}^k/(\tilde{d}^{k-1})^T \tilde{d}^{k-1})], \qquad 1 \leq \beta \leq 2, \tag{43}$$

and $\tilde{d}^{k-1}$ is the direction applied at the previous iteration.

For the above modified surrogate subgradient method, Proposition 4.1 and Theorem 4.1 still hold. In addition to convergence, it can be proved that the modified surrogate subgradient directions are better than the surrogate subgradient directions.

Theorem 5.1. The modified surrogate subgradient direction d^k satisfies

$$(\lambda^* - \lambda^k)^T \bar{d}^k / \|\bar{d}^k\| \geq (\lambda^* - \lambda^k)^T \tilde{g}^k / \|\tilde{g}^k\|. \tag{44}$$

Proof. The proof is similar to the proof presented in Camerini *et al.* Ref. 9 and is, therefore, omitted. □

Theorem 5.1 states that the modified surrogate subgradient direction forms a smaller or equal angle with the direction toward the optimal λ^* than does the surrogate subgradient direction.

6. Numerical Testing

Since no special features of integer programming are used in establishing the surrogate subgradient method and the modified surrogate subgradient method, these methods can be used to solve general separable linear or nonlinear programming problems. In this section, a small nonlinear programming problem is tested first. The convergence speed and the quality of the directions of the surrogate methods are compared with those of the gradient method. These methods are then used to solve several job shop scheduling problems.

6.1. Nonlinear Programming Example. Consider the following nonlinear programming problem:

$$\min \quad 0.5x_1^2 + 0.1x_2^2, \tag{45}$$

$$\text{s.t.} \quad x_1 - 0.2x_2 \geq 48, \tag{46}$$

$$5x_1 + x_2 \geq 250, \tag{47}$$

$$x_1, x_2 \in R. \tag{48}$$

This separable problem is solved by Lagrangian relaxation with two multipliers and two subproblems (one decision variable for each subproblem). The gradient method, the surrogate gradient method, and the modified surrogate gradient method are used to maximize the Lagrangian dual. In the surrogate methods, only one decision variable is optimized to obtain a direction. The multipliers λ_1 and λ_2 are updated 15 iterations in the gradient method and

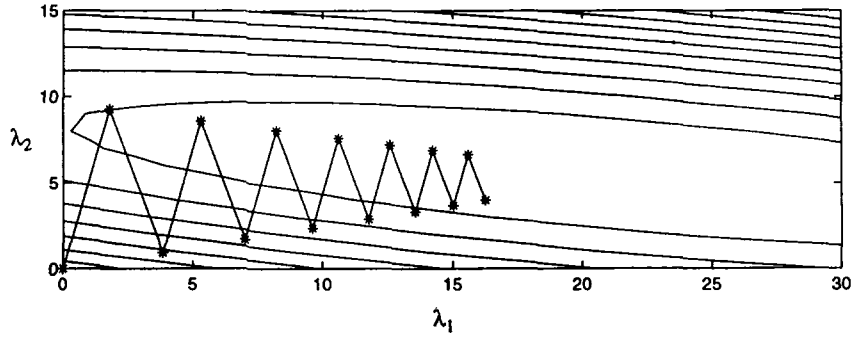

Fig. 1. Trajectories of multipliers by using the gradient method.

30 iterations in the surrogate methods (each subproblem is solved 15 times as in the gradient method). The trajectories of λ_1 and λ_2 for the three methods are shown in Figs. 1–3.

Since not all the subproblems are solved in the surrogate methods, surrogate directions are easier to obtain than gradient directions. In addition, as shown in the above figures, the gradient directions often zigzag from one iteration to the next. In this case, the surrogate gradient directions may in fact be better directions with less zigzagging, and the modified surrogate gradient directions may even further reduce zigzagging.

6.2. Job Shop Scheduling Example. The following integer programming example demonstrates how the surrogate subgradient method can be used to solve a job shop scheduling problem. In the basic job shop formulation, each part has its due date, priority (reflected by a weighting factor),

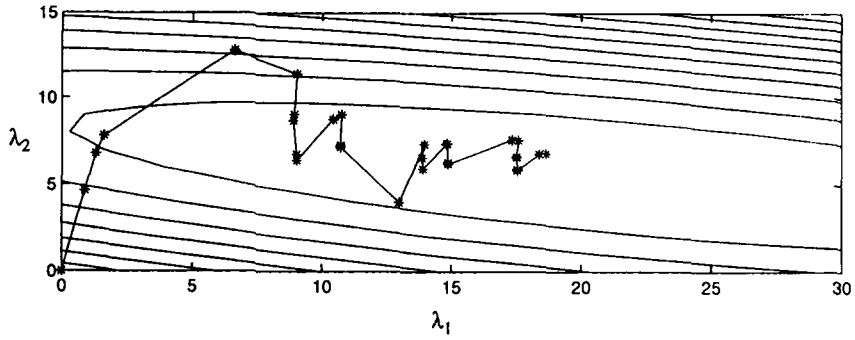

Fig. 2. Trajectories of multipliers by using the surrogate gradient method.

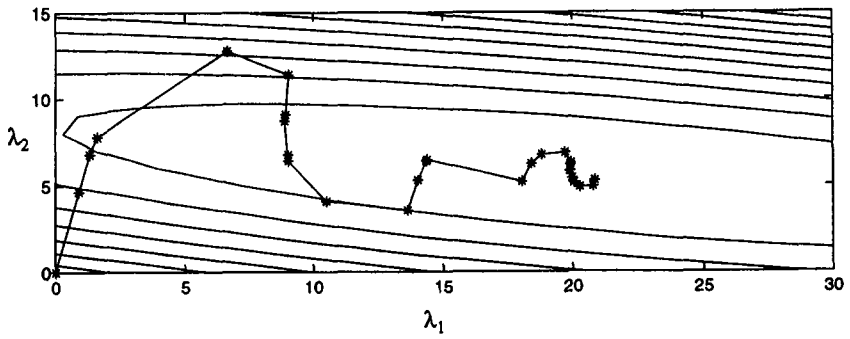

Fig. 3. Trajectories of multipliers by using the modified surrogate gradient method.

and requires a series of operations for completion. Each operation is to be performed on a machine of a specified type for a given period of time. The processing may start only after its preceding operations have been completed, satisfying the operation precedence constraint. Furthermore, the number of operations on a machine type at any time may not exceed the number of machines available, satisfying the machine capacity constraints. Through appropriate selection of decision variables, these constraints are formulated in separable forms (Refs. 4 and 10). The goals of on-time deliveries and low inventory are modeled as penalties on delivery tardiness and on releasing raw materials too early. The problem is to determine the operation beginning times so that the objective function is minimized. The key feature of this formulation is its separability.

For this NP-hard problem, that has prohibitive computational requirements to obtain an optimal solution, a combined Lagrangian relaxation and heuristic approach is used here to obtain near-optimal solutions with quantifiable quality. In the method, hard machine capacity constraints are first relaxed by introduction of a cost for using a machine. The cost at a particular time slot is the Lagrange multiplier, or the shadow price of the machine type in the economics literature. Since the original problem is separable, the relaxed problem can be decomposed into many smaller subproblems, one for each part. Given the set of multipliers or prices, each part is scheduled on the required machine so as to minimize its cost under operations precedence constraints. These subproblems are not NP-hard and can be solved efficiently by using dynamic programming. The multipliers are then adjusted iteratively, and the dual function is maximized. At the termination of such updating iterations, simple heuristics are applied to adjust the

subproblem solutions to remove any infeasibility and form a feasible schedule satisfying all the constraints. Furthermore, the dual cost is a lower bound on the optimal cost; therefore, the quality of the solutions can be evaluated by comparing its cost with the maximum of the dual costs obtained.

For practical job shop scheduling problems, the size can be very large, and the solution of the subproblems by dynamic programming takes the majority of the computation time. For example, it takes more than 70% of the total CPU time to solve the subproblems by using dynamic programming for a problem with 82 parts (Ref. 4). It is also reported that the subgradient method tends to zigzag and leads to slow convergence. Thus, it takes a long time to solve a large problem by using the subgradient method, and a better algorithm is required to improve the overall performance.

Both the subgradient method (SG) and the surrogate subgradient method (SSG) are used here to maximize the dual function. In the surrogate subgradient method, the interleaved idea is used and only one subproblem is solved in each iteration to obtain the surrogate subgradient direction. The algorithm is stopped if the CPU time is more than 20 min or the duality gap is less than 10%. Testing results are presented in Table 1.

The testing results show a significant performance gain by using the surrogate subgradient method. This is because a surrogate subgradient direction can be obtained without solving all the subproblems; thus, much effort is saved to get a direction for problems of large size. Furthermore, when the subgradient direction is zigzagging, the surrogate subgradient direction is smoother and may yield a better result.

Table 1. Comparison of SSG and SG algorithms for job shop scheduling problems.

Primal dimensions MT/M/P/O/D[a]	Optimization method	Dual cost[b]	Primal cost	Duality gap (%)	CPU time[c]
11/16/18/159/1287	SSG	12895	14236	10.4	89
	SG	12929	14222	10.0	321
8/14/82/752/1480	SSG	33375	37413	12.1	1200
	SG	31118	37653	21.0	1200
22/87/100/633/1628	SSG	106147	117081	10.3	667
	SG	102525	117768	14.9	1200

[a]The notation MT/M/P/O/D provides the number of machine types (MT), number of machines (M), number of parts (P), number of operations (O), and number of Lagrangian multipliers (D).

[b]In order to compare with SG, all the subproblems are solved optimally and a true dual is obtained at the last iteration of SSG. Based on the testing results, the surrogate dual is not far from the dual.

[c]CPU time is in seconds on a SUN Ultra1 workstation.

7. Conclusions

The key idea of the surrogate gradient method is that an approximate optimization can be used to obtain a proper surrogate gradient direction. This method includes the interleaved subgradient method as a special case and provides a framework allowing creative variations. Compared with methods such as the conjugate method or bundle method, which take much effort to find a better direction, the surrogate gradient method saves effort in obtaining a direction and provides a new approach for speeding up computation. The directions obtained are also smooth for large problems, leading to significant performance gain.

References

1. GEOFFRION, A. M., *Lagrangian Relaxation for Integer Programming*, Mathematical Programming Study, North Holland, Amsterdam, Holland, Vol. 2, pp. 82–114, 1974.
2. HIRIART-URRUTY, J. B., and LEMARECHAL, C., *Convex Analysis and Minimization Algorithms, Vols. 1–2*, Springer Verlag, Berlin, Germany, 1993.
3. WANG, J., LUH, P. B., ZHAO, X., and WANG, J., *An Optimization-Based Algorithm for Job Shop Scheduling*, Sadhana, Vol. 22, Part 2, pp. 241–256, 1997.
4. KASKAVELIS, C. A., and CARAMANIS, M. C., *Efficient Lagrangian Relaxation Algorithms for Real-Life-Size Job-Shop Scheduling Problems*, Working Paper, Department of Manufacturing Engineering, Boston University, Boston, Massachusetts, 1995.
5. LIU, Y., and STOREY, C., *Efficient Generalized Conjugate Gradient Algorithms, Part 1: Theory*, Journal of Optimization Theory and Applications, Vol. 69, pp. 129–137, 1991.
6. NEMHAUSER, G., and WOLSEY, L., *Integer and Combinatorial Optimization*, John Wiley and Sons, New York, New York, 1988.
7. BERTSEKAS, D. P., *Nonlinear Programming*, Athena Scientific, Belmont, Massachusetts, pp. 594–595, 1995.
8. SARIN, S., KARWAN, M. H., and RARDIN, R. L., *Surrogate Duality in a Branch-and-Bound Procedure for Integer Programming*, European Journal of Operational Research, Vol. 33, pp. 326–333, 1988.
9. CAMERINI, P., FRATTA, L., and MAFFIOLI, F., *On Improving Relaxation Methods by Modified Gradient Techniques*, Mathematical Programming Study, North Holland, Amsterdam, Holland, Vol. 3, pp. 26–34, 1975.
10. LUH, P. B., and HOITOMI, D. J., *Scheduling of Manufacturing Systems Using the Lagrangian Relaxation Technique*, IEEE Transactions on Automatic Control, Vol. 38, pp. 1066–1079, 1993.