JOURNAL OF OPTIMIZATION THEORY AND APPLICATIONS: Vol. 113, No. 2, pp. 373–397, May 2002 (©2002)

New Bundle Methods for Solving Lagrangian Relaxation Dual Problems[1]

X. ZHAO[2] AND P. B. LUH[3]

Communicated by W. B. Gong

Abstract. Bundle methods have been used frequently to solve nonsmooth optimization problems. In these methods, subgradient directions from past iterations are accumulated in a bundle, and a trial direction is obtained by performing quadratic programming based on the information contained in the bundle. A line search is then performed along the trial direction, generating a serious step if the function value is improved by ϵ or a null step otherwise. Bundle methods have been used to maximize the nonsmooth dual function in Lagrangian relaxation for integer optimization problems, where the subgradients are obtained by minimizing the performance index of the relaxed problem. This paper improves bundle methods by making good use of near-minimum solutions that are obtained while solving the relaxed problem. The bundle information is thus enriched, leading to better search directions and less number of null steps. Furthermore, a simplified bundle method is developed, where a fuzzy rule is used to combine linearly directions from near-minimum solutions, replacing quadratic programming and line search. When the simplified bundle method is specialized to an important class of problems where the relaxed problem can be solved by using dynamic programming, fuzzy dynamic programming is developed to obtain efficiently near-optimal solutions and their weights for the linear combination. This method is then applied to job shop scheduling problems, leading to better performance than previously reported in the literature.

Key Words. Lagrangian relaxation, bundle methods, nonsmooth optimization, manufacturing scheduling.

[1]This work was supported in part by the National Science Foundation under Grants DMI–9800037 and ECS-9726577.
[2]Member of the Advanced Technical Staff, I2 Technologies, Cambridge, Massachusetts.
[3]Professor, Department of Electrical and Computer Engineering, School of Engineering, University of Connecticut, Storrs, Connecticut.

1. Introduction

Integer optimization problems are generally difficult to solve because of their inherent combinatorial complexity, and Lagrangian relaxation has been a powerful approach to obtain near-optimal solutions. In Lagrangian relaxation, certain constraints are first relaxed through the introduction of Lagrangian multipliers. The relaxed problem is easier than the original one and can be solved efficiently if it belongs to Class P. Multipliers are then iteratively adjusted based on the level of constraint violation. The dual function is maximized in this multiplier updating process, and the values of the dual function serve as lower bounds to the optimal feasible cost (Ref. 1). At the termination of such updating iterations, simple heuristics are applied to adjust the relaxed problem solutions to form a feasible result satisfying all the constraints.

A major challenge in Lagrangian relaxation is to maximize effectively the dual function, which is concave, piecewise linear, and consists of many facets. The subgradient method is commonly used, where a subgradient can be obtained by minimizing the relaxed problem, and the multipliers are updated along this direction (Ref. 2). However, the multipliers often zigzag across ridges (intersections of two or more facets) of the dual function, and require many iterations to reach an optimum. Recently, the surrogate subgradient method (SSG) was developed, where only an approximate optimization of the performance index of the relaxed problem is needed to obtain a proper surrogate subgradient direction to update the multipliers (Refs. 3–4). Compared with methods that take effort to find good directions, this method obtains directions with much less effort and provides a new approach for solving large problems.

Bundle methods represent a quite different approach for nonsmooth optimization, and aim to find an ϵ-ascent direction along which a function value can increase by at least ϵ (Ref. 5). In these methods, subgradients from past iterations are accumulated in a bundle, and a trial direction is obtained by quadratic programming based on the bundle information. Line search is then performed along the trial direction, generating a serious step if the function value is improved by ϵ or a null step otherwise. Since a serious step is generated at the cost of quadratic programming and line search, with the possibility of having multiple null steps in-between two serious steps, much computation is required.

The Lagrangian relaxation framework and the methods for solving the nonsmooth dual problem as discussed above will be presented briefly in Section 2. Based on the insights obtained from these methods, ϵ-minimum solutions of the relaxed problem and ϵ-surrogate subgradients are introduced, and the improved bundle method is developed in Section 3. The key idea is to make good use of all the information obtained during the minimization of the performance index of the relaxed problem, not just the minimum solution but also near-minimum solutions. The bundle information is thus enriched, leading to better search directions and less number of null steps.

To reduce the computational requirements in solving large problems, a simplified bundle method is developed in Section 4 to decrease the distance to the optimal solution instead of requiring the function value to be increased by ϵ. In this way, the quadratic programming and line search required by traditional bundle methods are no longer necessary, and a fuzzy rule is established to combine linearly the subgradients associated with the near-optimal solutions of the relaxed problem. When the simplified bundle method is specialized to an important class of problems where the relaxed problem can be solved by using dynamic programming (DP), fuzzy dynamic programming (FZDP) is developed to obtain efficiently near-optimal solutions and their weights for the linear combination. The convergence of the method is proved, and the testing result presented in Section 5 on a benchmark problem shows that the simplified bundle method generates a similar result but with less computational requirements as compared to results previously obtained by using traditional bundle methods. Fuzzy dynamic programming is then applied to job shop scheduling problems in Section 6, where the complexity of FZDP can be reduced further by exploiting the special structure of the relaxed problem. Testing results show that the simplified bundle/FZDP method leads to better performance as compared with result previously reported in the literature. This method is generic for separable integer or mixed integer optimization problems beyond job shop scheduling, and provides a powerful approach to solve large-scale dual problems within the Lagrangian relaxation framework.

2. Problem Description and Formulation

The Lagrangian relaxation framework and several methods to solve the dual problem are introduced in this section.

2.1. Lagrangian Relaxation. An integer optimization problem can be described as follows:

$$\text{(IP)}\ \min_{x} \quad J(x), \tag{1}$$

$$\text{s.t.} \quad g(x) \leq 0 \quad \text{and} \quad x \in X. \tag{2}$$

Here, x is an $n \times 1$ decision variable belonging to the integer space $X = Z^n$. The constraint $g(x)$ is an $m \times 1$ function, and the cost $J(x)$ is a scalar function. In the Lagrangian relaxation approach, the constraints $g(x) \leq 0$ are relaxed by introducing the $m \times 1$ multiplier vector λ and the Lagrangian function

$$\tilde{L}(x, \lambda) \equiv J(x) + \lambda^T g(x). \tag{3}$$

The relaxed problem is to minimize $\tilde{L}(x, \lambda)$ over X, resulting in the concave nonsmooth dual function $L(\lambda)$,

$$L(\lambda) \equiv \min_{x \in X} \tilde{L}(x, \lambda). \tag{4}$$

The dual problem is to maximize the dual function (Ref. 6),

$$\max_{\lambda \geq 0} L(\lambda), \tag{5}$$

with the optimal dual solution denoted by λ^* and the optimal dual value denoted by $L^* = L(\lambda^*)$. In most cases, the optimization in (5) is performed iteratively, and at the termination of such iterations, a simple heuristics is applied to adjust the relaxed problem solutions so as to form a feasible result satisfying all the constraints.

2.2. Methods for the Dual Problem.

Subgradient Method. Since the dual problem is nondifferentiable for integer optimization problems, the subgradient method is commonly used to maximize the dual function. In order to get a subgradient direction, a minimum solution for the relaxed problem is obtained,

$$x^k = \arg\min_{x \in X} \tilde{L}(\lambda^k, x), \tag{6}$$

where k is the iteration index. A subgradient is then calculated based on this minimum solution,

$$g^k = g(x^k). \tag{7}$$

In the subgradient method, the multipliers are updated along the subgradient direction,

$$\lambda^{k+1} = \lambda^k + s^k g^k, \tag{8}$$

where the stepsize s^k satisfies

$$0 < s^k < 2\,(L^* - L^k)/\|g^k\|^2. \tag{9}$$

The method requires the minimization of the performance index of the relaxed problem to obtain a subgradient and will reduce the distance to the optimal point step-by-step (Ref. 7). However, it may suffer from the difficulty of zigzagging across ridges of the dual function (Ref. 2).

Surrogate Subgradient Method. To overcome the above-mentioned difficulties and to solve more efficiently large integer optimization problems, the surrogate subgradient method was recently developed. It provides a new approach to speed up convergence by reducing the effort to obtain a direction. Instead of minimizing the performance index of the relaxed problem to obtain an optimal solution, only an approximate minimization is performed where the new iterate $x(\lambda^k)$ should satisfy the following condition for the given set of multipliers λ^k (Ref. 3):

$$x(\lambda^k) \in \{x | \tilde{L}(\lambda^k, x) < \tilde{L}(\lambda^k, x^{k-1}), x \in X\}. \tag{10}$$

The surrogate subgradient direction $\tilde{g}^k$ is calculated based on an approximate solution x^k,

$$\tilde{g}^k = g(x^k), \tag{11}$$

and the multipliers are updated along the surrogate subgradient direction. It has been proved that the distance to the optimal point is reduced step-by-step under the following stepsize rule:

$$0 < s^k < (L^* - \tilde{L}^k)/\|\tilde{g}^k\|^2. \tag{12}$$

Note that the upper bound in (12) for the surrogate subgradient method is half of the bound in (9) for the gradient method, indicating that, without the accurate x^k, the stepsizing rule should be more conservative. Nevertheless, with the approximate minimization employed in the relaxed problem, the computational complexity to obtain a direction can be much reduced. For example, when the relaxed problem can be decomposed into N subproblems, only one subproblem needs to be solved to satisfy (10). This specialized version is the interleaved idea of Ref. 4, and the effort to obtain a direction is $1/N$ of that required by a traditional subgradient method.

Bundle Method. The bundle method has been a powerful approach for maximizing nonsmooth concave functions (Ref. 5). It employs a concept called the ϵ-subdifferential, defined as

$$\partial_\epsilon L(\lambda) \equiv \{g \in R^m | L(\bar{\lambda}) \leq L(\lambda) + \langle g, \bar{\lambda} - \lambda \rangle + \epsilon, \forall \bar{\lambda} \in R^m\}. \tag{13}$$

Elements in $\partial_\epsilon L(\lambda)$ are called ϵ-subgradients. Correspondingly, the ϵ-directional derivative along the direction d at λ is defined as

$$L'_\epsilon(\lambda, d) \equiv \sup_{t>0} [L(\lambda + td) - L(\lambda) - \epsilon]/t. \tag{14}$$

It has been shown that

$$L'_\epsilon(\lambda, d) = \inf_{g\in\partial_\epsilon L(\lambda)} g'd. \tag{15}$$

From (15), if a direction d can be found such that $L'_\epsilon(\lambda, d) > 0$, then the dual cost can be increased by at least ϵ. Therefore, it is desirable to select a search direction d^*such that the directional derivative is maximized, i.e.,

$$\begin{aligned} d^* &= \arg\{\max_{\|d\|=1} L'_\epsilon(\lambda, d)\} \\ &= \arg\{\max_{\|d\|=1} \inf_{g\in\partial_\epsilon L(\lambda)} g'd\} \\ &= \arg\{\inf_{g\in\partial_\epsilon L(\lambda)} \max_{\|d\|=1} g'd\} \\ &= \arg\{\inf_{g\in\partial_\epsilon L(\lambda)} \|g\|\}. \end{aligned} \tag{16}$$

Therefore, this d^* is the ϵ-subgradient with the smallest norm.

Generally, since the ϵ-subdifferential is very difficult to obtain, the idea of the bundle method is to accumulate subgradients of the past iterates in a bundle

$$B = \{g_1, g_2, \ldots, g_b\}$$

and to approximate $\partial_\epsilon L(\lambda)$ by the convex hull of the bundle elements,

$$P_b = \left\{g \,|\, g = \sum_{i=1}^{b} \alpha_i g_i, g_i \in B, 0 \leq \alpha_i, \sum_{i=1}^{b} \alpha_i = 1, \sum_{i=1}^{b} \alpha_i e_i \leq \epsilon\right\}, \tag{17}$$

where e_i is the linearization error for element i,

$$e_i = L(\lambda_i) + \langle g_i, \lambda - \lambda_i\rangle - L(\lambda). \tag{18}$$

A direction in P_b that has the smallest norm is obtained by using quadratic programming, and a line search is then performed along this trial direction. If a point in the trial direction leads to an ϵ-ascent of the function value, λ^k is updated to the new point, resulting in a serious step. If P_b is not adequate enough to approximate $\partial_\epsilon L(\lambda)$, the trial direction may not be an ϵ-ascent direction. In this case, λ^k is not updated and the new point is added to the bundle, resulting in a null step. Null steps generate more subgradients near λ^k so that P_b becomes closer to $\partial_\epsilon L(\lambda)$; however, several null steps may be required before a serious step is obtained.

The approach described above is sometimes referred to as the dual form of the bundle methods. There are also primal forms of the bundle methods derived from stabilized cutting-plane methods (Ref. 8). Bundle methods have been used to maximize nonsmooth Lagrangian dual functions so as to provide better directions than those of the subgradient method (Ref. 9). The quadratic programming and line search involved, however, require much computation. Recently, second-order information was explored for bundle methods so as to improve the convergence rate at the cost of even more computation (Ref. 10).

3. Improved Bundle Methods

Bundle methods are developed based on the assumption that one subgradient can be obtained for a given λ. When this method is applied to maximize the dual function within the Lagrangian relaxation framework, the dual value and the subgradient are obtained by minimizing the performance index of the relaxed problem (6). However, solving the relaxed problem is problem dependent and can be quite time consuming. For example, it has been reported that around 80% of the total CPU time is spent on solving the relaxed problem in job shop scheduling (Ref. 11). In addition, multiple null steps may be needed to accumulate neighborhood subgradients if P_b is not a good approximation of $\partial_\epsilon L(\lambda)$. Since the relaxed problem must be solved at least once for every null step, it is desirable to approximate $\partial_\epsilon L(\lambda)$ with as small number of null steps as possible. Within the Lagrangian relaxation framework, it will be shown that near-optimal solutions of the relaxed problem contain valuable information, which can be used to improve the approximation of $\partial_\epsilon L(\lambda)$, thereby reducing the number of null steps needed.

To be precise, define the set of ϵ-optimal solutions for the relaxed problem as

$$X_Z(\lambda, \epsilon) \equiv \{x_i \mid \tilde{L}(\lambda, x_i) - L(\lambda) \leq \epsilon,\ x_i \in Z^n\}, \tag{19}$$

where x_i is an ϵ-optimal solution. Usually, these ϵ-optimal solutions are byproducts when the relaxed problem is solved and can be obtained without much effort. Accordingly, the ϵ-surrogate subdifferential is defined as

$$\partial \tilde{L}_\epsilon(\lambda) \equiv \left\{ \sum_i \alpha_i g(x_i) \mid x_i \in X_Z(\lambda, \epsilon), \text{ and } \alpha_i > 0, \sum_i \alpha_i = 1 \right\}. \tag{20}$$

In the following theorem, it will be shown that ϵ-surrogate subgradients belong to ϵ-subdifferentials.

Theorem 3.1. Any ϵ-surrogate subgradient belongs to the ϵ-subdifferential, i.e.,

$$\partial \tilde{L}_\epsilon(\lambda) \subset \partial_\epsilon L(\lambda). \tag{21}$$

Proof. Based on the definition of (19), an ϵ-optimal solution x_i satisfies

$$\tilde{L}(\lambda, x_i) - L(\lambda) \leq \epsilon. \tag{22}$$

From the definition of the dual function in (4), we have that, for any $\bar{\lambda}$,

$$L(\bar{\lambda}) = \min_{x \in Z^n} \{J(x) + \bar{\lambda}^T g(x)\} \leq J(x_i) + \bar{\lambda}^T g(x_i). \tag{23}$$

From the definition of the relaxed problem in (3), we have

$$\tilde{L}(\lambda, x_i) = J(x_i) + \lambda^T g(x_i) = J(x_i) + \bar{\lambda}^T g(x_i) - (\bar{\lambda} - \lambda)^T g(x_i), \tag{24}$$

which can be rewritten as

$$J(x_i) + \bar{\lambda}^T g(x_i) = \tilde{L}(\lambda, x_i) + (\bar{\lambda} - \lambda)^T g(x_i). \tag{25}$$

Combining (22), (23), (25), we have

$$L(\bar{\lambda}) \leq \tilde{L}(\lambda, x_i) + (\bar{\lambda} - \lambda)^T g(x_i) \leq L(\lambda) + (\bar{\lambda} - \lambda)^T g(x_i) + \epsilon. \tag{26}$$

Thus, the direction $g(x_i)$ related to an ϵ-optimal solution x_i satisfies

$$L(\bar{\lambda}) \leq L(\lambda) + \langle g(x_i), \bar{\lambda} - \lambda \rangle + \epsilon, \qquad \forall \bar{\lambda}. \tag{27}$$

Based on the definition of $\partial_\epsilon L(\lambda)$ in (13),

$$g(x_i) \in \partial_\epsilon L(\lambda);$$

consequently,

$$g = \sum_i \alpha_i g(x_i) \in \partial L_\epsilon(\lambda).$$ □

Theorem 3.1 states that ϵ-optimal solutions provide valuable information and that the related ϵ-surrogate subgradients belong to $\partial_\epsilon L(\lambda)$. Since the bundle method uses P_b to approximate $\partial_\epsilon L(\lambda)$, $\partial \tilde{L}_\epsilon(\lambda)$ can be added to P_b to obtain a better approximation. In fact, for convex problems with real decision variables (i.e., $x \in R^n$ as opposed to $x \in Z^n$) and linear constraints, it can be proved that P_b itself is contained in $\partial \tilde{L}_\epsilon(\lambda)$.

Theorem 3.2. For a convex problem with real decision variables and linear constraints,

$$P_b \subset \partial \tilde{L}_\epsilon(\lambda). \tag{28}$$

Proof. From the definition of P_b in (17), given any $g \in P_b$, we have

$$g = \sum_{i=1}^{b} \alpha_i g_i, \qquad \sum_{i=1}^{b} \alpha_i = 1, \tag{29}$$

$$\sum_{i=1}^{b} \alpha_i e_i \leq \epsilon. \tag{30}$$

For each bundle element g_i, there is a corresponding λ_i, and the associated minimum solution x_i satisfies

$$g_i = g(x_i), \tag{31}$$

$$L(\lambda_i) = J(x_i) + \langle g_i, \lambda_i \rangle. \tag{32}$$

According to (18), we have

$$e_i = L(\lambda_i) + \langle g_i, \lambda - \lambda_i \rangle - L(\lambda). \tag{33}$$

Combining (32) and (33), we have

$$\begin{aligned} e_i &= J(x_i) + \langle g_i, \lambda_i \rangle + \langle g_i, \lambda - \lambda_i \rangle - L(\lambda) \\ &= J(x_i) + \langle g_i, \lambda \rangle - L(\lambda). \end{aligned} \tag{34}$$

This can be rewritten as

$$J(x_i) + \langle g_i, \lambda \rangle = L(\lambda) + e_i. \tag{35}$$

Since $J(x)$ is convex and $g(x)$ is linear, the following is true for $\sum_i \alpha_i x_i$:

$$\begin{aligned} \tilde{L}\left(\sum_i \alpha_i x_i, \lambda\right) &= J\left(\sum_i \alpha_i x_i\right) + g\left(\sum_i \alpha_i x_i\right)^T \lambda \\ &\leq \sum_i \alpha_i J(x_i) + \left\langle \sum_i \alpha_i g_i, \lambda \right\rangle. \end{aligned} \tag{36}$$

Combined with (35) and (30), we have

$$\tilde{L}\left(\sum_i \alpha_i x_i, \lambda\right) \leq \sum_i \alpha_i (J(x_i) + \langle g_i, \lambda \rangle) = \sum_i \alpha_i (L(\lambda) + e_i) \leq L(\lambda) + \epsilon. \tag{37}$$

For the case with real decision variables, the set of ϵ-optimal solution is defined as

$$X_R(\lambda, \epsilon) \equiv \{x | \tilde{L}(\lambda, x) - L(\lambda) \leq \epsilon, x \in R^n\}, \tag{38}$$

and the corresponding ϵ-surrogate subdifferential is

$$\partial \tilde{L}_\epsilon(\lambda) \equiv \{g(x), \forall x \in X_R(\lambda, \epsilon)\}. \tag{39}$$

From (37),

$$\sum_i \alpha_i x_i \in X_R(\lambda, \epsilon), \qquad g = \sum_i \alpha_i g_i = g\left(\sum_i \alpha_i x_i\right) \in \partial \tilde{L}_\epsilon(\lambda);$$

therefore, (28) is proved. □

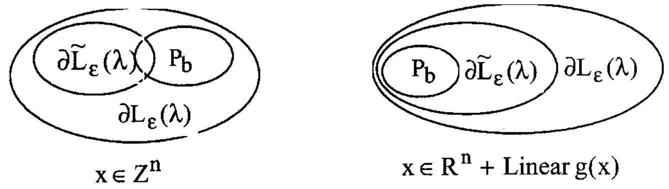

Figure 1. Relationship among bundle P_b, ϵ-surrogate subdifferential $\partial\tilde{L}_\epsilon(\lambda)$, and ϵ-subdifferential $\partial L_\epsilon(\lambda)$.

From the above theorems, the relationship among the bundle P_b, ϵ-surrogate subdifferential $\partial\tilde{L}_\epsilon(\lambda)$, and ϵ-subdifferential $\partial L_\epsilon(\lambda)$ is summarized in Fig. 1. Since it is not guaranteed that

$$\sum_i \alpha_i x_i \in X_Z(\lambda, \epsilon),$$

Theorem 3.2 is not true for general integer optimization problems.

For integer optimization, the dual function is a piecewise linear concave function with many facets, and each facet corresponds to a solution of the relaxed problem. A near-minimum solution for a given λ^k is associated with a facet that is close to the facet corresponding to the minimum solution as illustrated in Fig. 2. In this figure, x^3 is the minimum solution given λ^k and x^2 is a near-minimum solution. Their corresponding facets are close to each other around λ^k, and the gradient associated with x^2 belongs to $\partial\tilde{L}_\epsilon(\lambda^k)$.

When P_b is not a good approximation of $\partial L_\epsilon(\lambda)$, the bundle methods accumulate subgradients from nearby facets with several null steps. Since most neighborhood subgradients are in fact already contained in $\partial\tilde{L}_\epsilon(\lambda)$, some null steps may not be necessary if the information in $\partial\tilde{L}_\epsilon(\lambda)$ is utilized fully. Therefore, by combining $\partial\tilde{L}_\epsilon(\lambda)$ with P_b, the bundle method can be enriched, leading to better trial directions and less number of null steps.

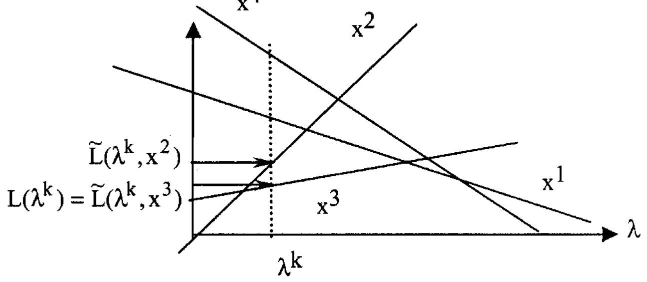

Figure 2. Near-minimum solution and nearby intersecting facet.

4. Simplified Bundle Method

4.1. Relaxed Convergence Condition. Compared with traditional bundle methods, the improved bundle method provides better directions by enriching the bundle with ϵ-surrogate subdifferential. Quadratic programming and line search, however, are still required to guarantee the algorithm convergence. This is because both methods try to improve the objective function by at least ϵ for each serious step. Instead of improving the function value, another approach is to reduce the distance between the current iterate to an optimal solution step-by-step. It will be shown in this section that the convergence conditions for such an algorithm will be much relaxed.

Given the current iterate λ, define the optimal direction to be the direction emanating from λ to λ^*, i.e., $g^*(\lambda) \equiv \lambda^* - \lambda$. We have the following theorem.

Theorem 4.1. Any direction $g \in P_b$ or $g \in \partial \tilde{L}_\epsilon(\lambda)$ is at an acute angle with the optimal direction if ϵ is sufficiently small; i.e.,

$$0 < [L^* - L(\lambda)]/2 < g^T(\lambda^* - \lambda), \tag{40}$$

if

$$\epsilon < [L^* - L(\lambda)]/2. \tag{41}$$

Proof. For any $g \in P_b$,

$$g = \sum_{i=1}^{b} \alpha_i g_i.$$

From (35), within the proof of Theorem 3.2, we have

$$\tilde{L}(\lambda, x_i) = J(x_i) + \langle g_i, \lambda \rangle = L(\lambda) + e_i, \tag{42}$$

where x_i is the minimum solution for λ_i associated with the bundle element g_i. Since a minimization is performed in deriving $L(\lambda)$, $\tilde{L}(\lambda, x_i)$ is greater than or equal to the dual; i.e.,

$$L(\lambda) \leq \tilde{L}(\lambda, x_i), \qquad \forall x_i \text{ and } \lambda. \tag{43}$$

The above is also true at λ^*, i.e.,

$$L^* = L(\lambda^*) \leq \tilde{L}(\lambda^*, x_i). \tag{44}$$

Then, from (42), we have

$$L^* - L(\lambda) - e_i = L^* - \tilde{L}(\lambda, x_i) \leq \tilde{L}(\lambda^*, x_i) - \tilde{L}(\lambda, x_i), \tag{45}$$

which from (3) can be written as

$$L^* - L(\lambda) - e_i \le g(x_i)^T(\lambda^* - \lambda). \tag{46}$$

Combining (30), (41), (46) one obtains

$$\begin{aligned} 0 &< [L^* - L(\lambda)]/2 < L^* - L(\lambda) - \epsilon \\ &< \sum_i \alpha_i (L^* - L(\lambda) - e_i) \\ &< \left(\sum_i \alpha_i g_i\right)^T (\lambda^* - \lambda). \end{aligned} \tag{47}$$

Equation (40) is thus proved for $g \in P_b$.

For any $g \in \partial \tilde{L}_\epsilon(\lambda)$,

$$g = \sum_i \alpha_i g(x_i),$$

where x_i is an ϵ-optimal solution satisfying

$$\tilde{L}(\lambda, x_i) - L(\lambda) \le \epsilon. \tag{48}$$

Given (39) and (41), we have

$$\tilde{L}(\lambda, x_i) \le \epsilon + L(\lambda) \langle [L^* - L(\lambda)]/2 + L(\lambda) = [L^* + L(\lambda)]/2. \tag{49}$$

From (41), (49), (4), we have

$$0 < [L^* - L(\lambda)]/2 \langle L^* - \tilde{L}(\lambda, x_i) \le \tilde{L}(\lambda^*, x_i) - \tilde{L}(\lambda, x_i), \tag{50}$$

which can be rewritten as

$$0 < [L^* - L(\lambda)]/2 \le g(x_i)^T(\lambda^* - \lambda); \tag{51}$$

therefore, (40) is proved for $g \in \partial \tilde{L}_\epsilon(\lambda)$. □

Theorem 4.1 states that any direction $g \in P_b$ or $g \in \partial \tilde{L}_\epsilon(\lambda)$ is at an acute angle with the direction pointing to λ^*, if ϵ is sufficiently small as illustrated in Fig. 3. In the following, it will be shown that the distance to the optimal λ^* can be reduced step-by-step.

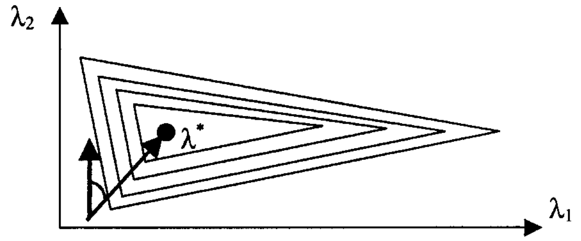

Figure 3. Surrogate subgradient forming an acute angle with the optimal direction.

Theorem 4.2. If the multipliers are updated as follows:

$$\lambda^{k+1} = \lambda^k + s^k g^k, \tag{52}$$

with

$$g^k \in \partial \tilde{L}_\epsilon(\lambda^k) \qquad \text{or} \qquad g^k \in P_b, \tag{53a}$$

and if

$$0 < s^k < (L^* - L(\lambda^k))/\|g^k\|^2, \tag{53b}$$

then the multipliers move closer to an optimal λ^* step-by-step; i.e.,

$$\|\lambda^* - \lambda^{k+1}\| < \|\lambda^* - \lambda^k\|, \qquad \text{for all } k. \tag{54}$$

Proof. From (52),

$$\|\lambda^* - \lambda^{k+1}\|^2 = \|\lambda^* - \lambda^k\|^2 - 2s^k(\lambda^* - \lambda^k)^T g^k + (s^k)^2\|g^k\|^2. \tag{55}$$

Combining with (40), the above yields [for brevity, we set $L^k \equiv L(\lambda^k)$]

$$\|\lambda^* - \lambda^{k+1}\|^2 \leq \|\lambda^* - \lambda^k\|^2 - s^k(L^* - L^k) + (s^k)^2\|g^k\|^2. \tag{56}$$

It can be rewritten as

$$\|\lambda^* - \lambda^{k+1}\|^2 \leq \|\lambda^* - \lambda^k\|^2 - s^k[(L^* - L^k) - s^k\|g^k\|^2]. \tag{57}$$

For the range of stepsizes in (53), the term in the pair of brackets is greater than zero; thus, (54) is proved. □

4.2. Simplified Bundle Method. Similar to the convergence proof of subgradient methods, the above theorems demonstrate that quadratic programming and line search are not necessary to guarantee algorithm convergence. Therefore, it is possible to develop simpler and more flexible algorithms. For example, we can use the stepsizing rule (53) instead of the line search. Furthermore, instead of obtaining the direction with the minimum norm by using quadratic programming, it will be much easier to get a direction with a small norm by using some sensible rules. In fact, any direction $g \in P_b$ or $g \in \partial \tilde{L}_\epsilon(\lambda)$ is eligible. One intuitively appealing idea is to form a fuzzy set of near-optimal solutions and define the membership $\bar{\alpha}_i$ of a near-optimal solution x_i to be the closeness of $\tilde{L}(x_i, \lambda)$ to $L(\lambda)$, e.g.,

$$\bar{\alpha}_i \equiv [L(\lambda) + \epsilon - \tilde{L}(x_i, \lambda)]/\epsilon, \qquad \text{with } x_i \in X_Z(\lambda, \epsilon), \tag{58a}$$

$$\bar{\alpha}_i \equiv 0, \qquad \text{otherwise}. \tag{58b}$$

A fuzzy gradient is then obtained by combining linearly the gradients of all the elements in the fuzzy set, with the weight of an element equal to its normalized fuzzy membership, i.e.,

$$g = \sum_i \alpha_i g(x_i), \qquad \text{with } \alpha_i \equiv \bar{\alpha}_i / \sum_i \bar{\alpha}_i. \tag{59}$$

Combining the ideas obtained in Sections 3 and 4, a simplified bundle method can be constructed. In this method, the directions $\partial \tilde{L}_\epsilon(\lambda)$ from the near-optimal solutions form the bundle, and a simple fuzzy rule such as (59) is used to obtain a search direction. For simplicity of computation, the stepsizing rule (53) is applied instead of a line search to update the multipliers.

4.3. Fuzzy Dynamic Programming. The above simplified bundle method provides a framework to utilize ϵ-optimal solutions of the relaxed problem, assuming that these solutions are available. In certain cases, it may not be easy to obtain ϵ-optimal solutions if the relaxed problem is complicated. In the following, we will present how to implement the fuzzy idea for an important class of problems where the relaxed problem can be solved by using dynamic programming (Ref. 12). This is done by exploiting the closed-loop nature of DP solutions where near-optimal solutions are obtainable without much additional effort than that needed for obtaining an optimal solution. To be precise, the standard DP is first introduced.

Dynamic Programming. Suppose that the relaxed problem is a discrete-time and discrete-state optimal control problem described by

$$x_{\{t\}+1} = f_t(x_t, u_t), \qquad \text{for } t = 0, \ldots, T-1,$$

with the initial state x_0 given. The objective function to be minimized is

$$L = g_T(x_T) + \sum_{t=0}^{T-1} g_t(x_t, u_t).$$

In the above, t is the stage or time index, x_t the state, and u_t the control that governs the state transition; $g_t(x_t, u_t)$ is the stagewise cost and $g_T(x_T)$ is the terminal cost. The structure of such a problem is illustrated in Fig. 4, where the optimal states are represented by gray nodes and the optimal controls are marked with weight 1.

A standard backward DP begins at the last stage with the terminal cost $V_T(x_T) = g_T(x_T)$ computed. It then moves backward to the previous stage. For a given state, the cost-to-go function for each possible control is calculated as

$$\tilde{V}_t(x_t, u_t) = g_t, (x_t, u_t) + V_{t+1}(f_t, (x_t, u_t)), \tag{60}$$

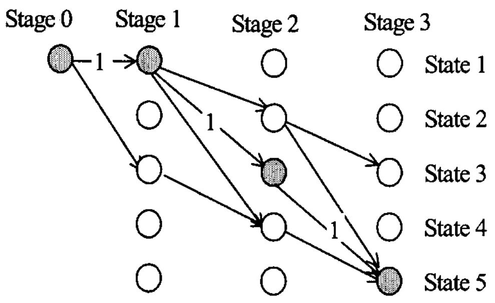

Figure 4. Structure of dynamic programming.

where the cost-to-go function $\tilde{V}_t(x_t, u_t)$ is the sum of the stagewise cost and the associated minimum cost-to-go starting from the next stage. The minimum cost-to-go function $V_t(x_t)$ is then obtained by minimizing $\tilde{V}_t(x_t, u_t)$ over the set of all possible controls $U_t(x_t)$, i.e.,

$$\begin{aligned} V_t(x_t) &= \min_{u_t \in U_t(x_t)} \tilde{V}_t(x_t, u_t) \\ &= \min_{u_t \in U_t(x_t)} \{g_t(x_t, u_t) + V_{t+1}(f_t(x_t, u_t))\}. \end{aligned} \tag{61}$$

The backward calculation proceeds from the last stage to the first stage, and the minimum cost-to-go at the first stage for the given initial state x_0 is the minimum cost. A forward sweep is then used to find the optimal path from the first stage for the given x_0 to the last stage as follows:

$$u_t^*(x_t) = \arg\min_{u_t \in U_t(x_t)} \tilde{V}_t(x_t, u_t), \qquad t = 0, \ldots, T-1, \tag{62}$$

$$x_{t+1}^* = f_t(x_t^*, u_t^*(x_t^*)), \qquad \text{with } x_0^* = x_0, t = 0, \ldots, T-1. \tag{63}$$

Based on $u_0^*(x_0)$ for the given x_0, it proceeds to the state at the next stage. This process continues until the last stage is reached.

Fuzzy Dynamic Programming. Usually, only one optimal path is obtained in DP, and these x_t^* and u_t^* are used to calculate a subgradient. However, in general, there could be many near-optimal or optimal paths, each associated with an ϵ-minimum solution or a minimum solution. In view of the closed-loop nature of DP, many ϵ-optimal paths can be obtained as byproducts when the DP procedure is performed without much additional efforts. Fuzzy dynamic programming (FZDP) is thus to assign weights to states and controls associated with these ϵ-optimal paths according to fuzzy rules, and to compute the corresponding fuzzy gradient to utilize fully the

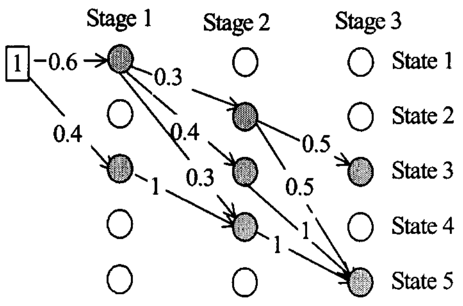

Figure 5. Structure of fuzzy dynamic programming.

information contained in near-optimal solutions. FZDP can be implemented efficiently by properly modifying the DP procedure as illustrated in Fig. 5, where proper weights are assigned to near-optimal controls.

In FZDP, the backward calculation of the minimum costs-to-go is the same as that in DP. The forward sweep begins at the initial state with the state weight $w(x_0) = 1$. Given a state with a positive weight, all possible transitions to the next stage are assigned control weights $w(x_t, u_t)$ following (58),

$$\bar{w}(x_t, u_t) = \begin{cases} [V(x_t) + \epsilon' - \tilde{V}(x_t, u_t)]/\epsilon', & \text{if } \tilde{V}(x_t, u_t) < V(x_t) + \epsilon', \\ 0, & \text{otherwise,} \end{cases} \tag{64}$$

$$w(x_t, u_t) = \bar{w}(x_t, u_t) \Big/ \sum_{U_t(x_t)} \bar{w}(x_t, u_t), \qquad t = 0, \ldots, T-1. \tag{65}$$

To obtain ϵ-minimum solutions for the relaxed problem, the parameter ϵ' in (64) is arbitrarily set to be $\epsilon' = \epsilon/T$, where ϵ is equally divided among the T stages. Given a state weight $w(x_{t-1})$ at stage $t-1$ and the associated control weight $w(u_{t-1})$, the state weight at stage t can be calculated as

$$w(x_t) = \sum (w(x_{t-1}, u_{t-1}) \cdot w(x_{t-1})), \tag{66a}$$

$$\text{s.t.} \quad x_t = f_{t-1}(x_{t-1}, u_{t-1}). \tag{66b}$$

The process then repeats until the terminal stage is reached. Consequently, states related to near-optimal paths are assigned with state weights $\{w(x_t)\}$ and controls with control weights $\{w(x_t, u_t)\}$. Such information can be used to calculate a fuzzy gradient following (59), and an example on job shop scheduling will be presented in Section 6.

Compared to DP, additional computations are required in the forward sweep of FZDP. Assuming that there are K_S states per stage and there are K_C controls per state, the complexity of the forward sweep in the worst case is $O(K_S K_C T)$, which is similar to the complexity of the backward calculation. Since there may be many states with zero weights, the additional computation is usually insignificant as compared to the time-consuming backward sweep.

In FZDP, any path with a positive weight is guaranteed to be an ϵ-optimal solution. However, not all ϵ-optimal solutions are obtained from the above process in view of the equal division of ϵ into T parts, one for each stage. The additional complexity to find all the remaining ϵ-optimal solutions in DP is nonetheless prohibitive. The key of FZDP is to consider a significant number of ϵ-optimal solutions and to reasonably assign their weights in a computationally efficient manner.

4.4. Comparison of Methods. For the simplified bundle method with fuzzy dynamic programming (SB/FZDP), the relaxed problem is solved by using FZDP for a given set of multipliers, and a fuzzy gradient is obtained to be the search direction by considering near-minimum solutions. The differences between SB/FZDP and other methods are highlighted in Table 1.

In a subgradient method, a minimum solution is selected with weight 1, and all other solutions are ignored. For the surrogate subgradient method, only a near-optimal solution obtained by approximate minimization is selected with weight 1. The bundle methods obtain a good search direction by combining the subgradients of many nearby points. However, the relaxed problem may have to be solved many times to obtain the needed subgradients for a serious step, and the combination of the individual subgradients is determined by quadratic programming. Therefore, the computational requirements for bundle methods are high. The improved bundle method is similar to the bundle methods, except that fewer null steps are

Table 1. Comparison of methods.

	SG	SSG	Bundle	Improved bundle	SB/FZDP
Relaxed problem minimized	Once	< Once[a]	Several times	< Several times[b]	Once
Neighborhood information considered	No	No	Yes	Yes	Yes
ϵ-optimal solutions used	Ignored	Utilized	Ignored	Utilized	Utilized
Line search	No	No	Yes	Yes	No
Weight assignment	N/A	N/A	QP	QP	Simple rule
Direction	Not good	Not good	Good	Best	Good
Complexity	Low	Lowest	Highest	High	Low

[a]Approximate minimization is performed once.
[b]Number of null steps for the improved bundle method is less than that for the bundle method.

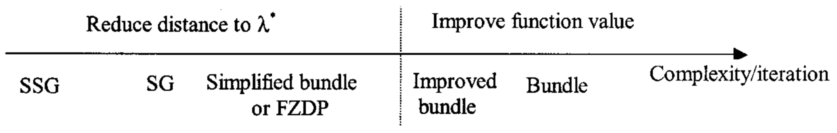

Figure 6. Complexity comparison of various methods.

required in view of the inclusion of the ϵ-surrogate subgradients. The simplified bundle method, on the contrary, does not require a line search nor quadratic programming. It makes good use of the subgradients of near-optimal solutions, which are generally available when the relaxed problem is solved, and these directions are combined linearly by using a simple rule. Computational requirements are thus relatively small. The complexity of one iteration of the above methods is depicted in the ascending order in Fig. 6.

However, the overall performance of a method is a tradeoff between the complexity per iteration versus the number of iterations required.

5. Numerical Testing

In this section, two examples are presented. Example 5.1 is to solve a simple dual problem to show that the simplified bundle method can reduce the solution zigzagging. Example 5.2 is a benchmark problem, and the performance of the simplified bundle method is compared with that of a traditional bundle method.

Example 5.1. As mentioned earlier, the subgradient directions often cause the multipliers to zigzag across sharp ridges. However, for the simplified bundle method since the direction is obtained by a weighted combination of the gradients of nearby facets, zigzagging is significantly reduced as illustrated below by maximizing the following piecewise linear dual function:

$$L(\lambda_1, \lambda_2) = \min\{-\lambda_1 - 3\lambda_2 + 300, -\lambda_1 + \lambda_2 + 1, 2\lambda_1 - \lambda_2 + 1\}. \tag{67}$$

The parameter ϵ for the simplified bundle method is set to be

$$\epsilon = (L^* - L)/2,$$

and

$$s = (L^* - L)/2\|\tilde{g}\|^2$$

is used following (53). The rules (58) and (59) are applied to obtain a combined direction based on $\partial \tilde{L}_\epsilon(\lambda)$. For the subgradient method,

$$s = (L^* - L)/2\|\tilde{g}\|^2$$

is also used. For both methods,

$$L^* = 25(11/12)$$

is assumed to be known within their stepsizing rule formulas.

The multiplier trajectories for the first ten iterations of both methods are shown in Fig. 7. Compared with the subgradient method, the simplified bundle method significantly reduces zigzagging. This is because, when the multipliers are close to a ridge, the search direction of the simplified bundle method is a combination of the gradients of nearby facets, generating a much smoother trajectory.

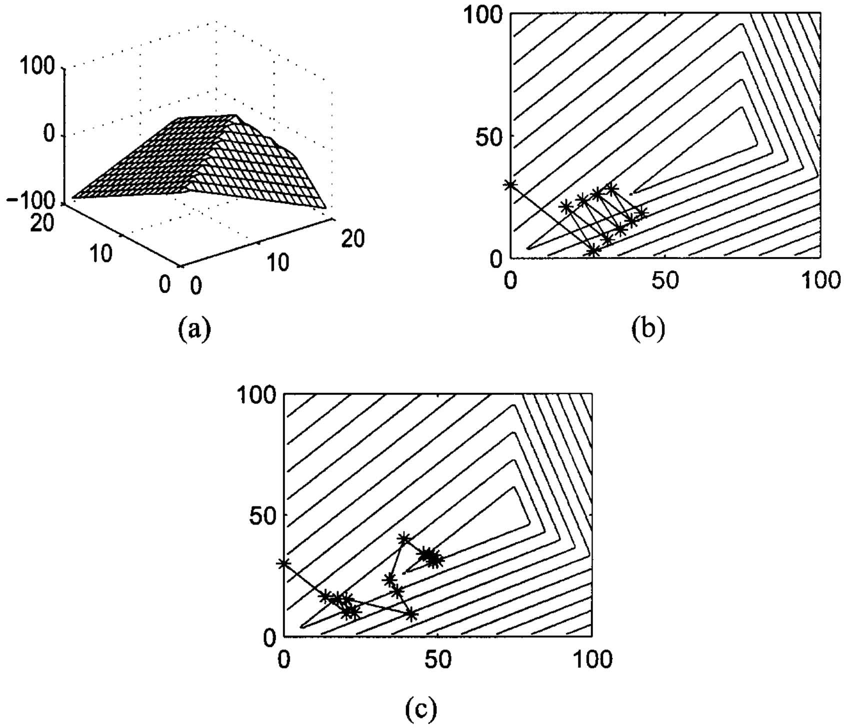

Figure 7. Reduction of solution zigzagging: (a) dual function, (b) trajectory obtained by using the subgradient method, and (c) trajectory obtained by using the simplified bundle method.

Example 5.2. Benchmark Problem. Bundle methods were developed for general nonsmooth optimization, and many benchmark problems have been reported in the literature. However, the improved bundle method and the simplified bundle method are developed within the Lagrangian relaxation framework so as to maximize the dual problem, which is a special class of nonsmooth optimization problems. Nevertheless, the concepts of near-optimal solutions and ϵ-surrogate subdifferential are generic, and the results in Sections 3 and 4 can be extended to linear min-max problems such as

$$\min f(x) \equiv \min_x \{\max (f_i(x): i = 1, \ldots, I)\}, \tag{68}$$

where $\{f_i(x)\}$ are linear functions. In the literature, the Goffin polyhedral problem is such a linear min-max benchmark problem,

$$f(x) = N \max \{x_i: i = 1, \cdots N\} - \sum_{i=1}^{N} x_i. \tag{69}$$

The problem dimension is $N = 50$, and the optimal cost is $f(x^*) = 0$. The initial point is given as

$$x_i^0 = i - (N+1)/2, \qquad \text{for } i = 1, \ldots, N,$$

with cost $f(x^0) = 1225$.

It is reported that bundle methods can get to the optimal solution with 51 iterations (34 serious steps and 17 null steps in Ref. 13). The results obtained by the subgradient method and the simplified bundle method are summarized in Table 2 for a few selected number of iteration indices. It can be seen that the performance of the simplified bundle method (SB) is much better than that of the subgradient method (SG). The performance of SB is also comparable with those obtained by bundle methods, having a similar number of iterations or function evaluations. However, the computation complexity in SB is much reduced as compared to traditional bundle methods, because neither quadratic programming nor line search is required.

Table 2. Testing results for the Goffin polyhedral problem.

Iterations		10	20	30	40	50
$f(x)$	SG	980.7	651.8	403.9	327.1	249.1
	SB	14.7839933	0.1600545	0.0014129	0.0000353	0.0000029

6. Job Shop Scheduling with Fuzzy Dynamic Programming

In this section, fuzzy dynamic programming is applied to job shop scheduling problems.

6.1. Problem Formulation and Solution Methodology. In a job shop, each part has its due date and weight or priority, and requires a series of operations for its 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. The number of operations assigned to a machine type may not exceed the number of machines available at any time, satisfying the machine capacity constraints. The problem is to determine the operation beginning times so that the total weighted part earliness and tardiness penalty is minimized. Through proper selection of the decision variables, these constraints are formulated in additive forms in Ref. 14. Unlike the prevalent formulations in the literature, the key feature here is its separability.

Within the Lagrangian relaxation framework, machine capacity constraints are relaxed by using Lagrange multipliers (capacity multipliers). For a given set of multipliers, the relaxed problem can be decomposed into decoupled part subproblems. Each subproblem represents the scheduling of a part so as to minimize its tardiness and earliness penalties and the costs for utilizing the machines (reflected by the values of the multipliers for the required machine types at scheduled time slots).

Each subproblem is a multistage optimization problem, and can be solved efficiently by using dynamic programming (DP) with polynomial complexity (Ref. 11). A typical DP structure is shown in Fig. 8. With stages

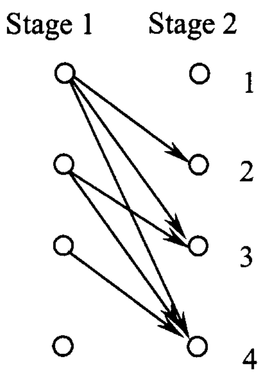

Figure 8. Dynamic programming for a part subproblem.

corresponding to operations and states corresponding to operation beginning times, the backward DP algorithm starts with the last stage and computes the tardiness penalties and machine utilization costs. As the algorithm moves backward, the cumulative costs of the individual states belonging to a particular stage are computed based on the stagewise costs and the minimal costs-to-go for the succeeding stage, subject to the allowable state transitions as delineated by the operation precedence constraints. The optimal subproblem cost is then obtained as the minimum of the cumulative costs at the first stage, and the optimal beginning times for the individual operations can be obtained by forward tracing the stages.

Each state node in Fig. 8 represents an operation beginning time and implies the utilization of a machine of a particular type for a specified period of time. Based on the optimal beginning times obtained from DP, the machine utilization by all the operations for each machine type at different time slots can be calculated. The subgradient is then a long vector of the difference between machine utilization and machine capacity for all machines and for all time periods. Iterative updating of the multipliers along a proper direction, repeated resolutions of subproblems, and the final heuristic adjustment of the subproblem solutions lead to the near-optimal solutions of the original problem. The cost of the feasible schedule J from the heuristic is an upper bound on the optimal feasible cost J^*. On the other hand, the optimal dual D^* is a lower bound on J^*. Since it is usually difficult to find J^* and D^*, the pseudoduality gap $(J\text{-}D)/D$ is often used as a measure of the quality of the feasible schedule, where D is the highest dual cost obtained over the iterations.

Only one optimal beginning time is usually obtained for each operation in a traditional DP. In FZDP, near-optimal beginning times are obtained, each with an associated state weight. The fuzzy machine utilization by the operation is calculated based on a weighted combination of these near-optimal beginning times. A fuzzy gradient can then be derived as the difference between the total fuzzy machine utilization and the machine capacity. Since near-optimal solutions are considered in FZDP, the resulting fuzzy gradient directions are much better than the subgradient directions. The complexity of FZDP can be reduced further by exploiting the special structure of the relaxed problem.

6.2. Testing Results. Several practical job shop scheduling problems were tested by using the simplified bundle/fuzzy dynamic programming (SB/FZDP) and the subgradient/dynamic programming method (SG/DP) within the Lagrangian relaxation framework on a Pentium 400 MHz PC. However, traditional bundle methods were not tested here because their significant computational requirements to obtain good search directions hinder their ability for handling large problems (Ref. 11). For both methods tested, the same heuristics was used to obtain feasible solutions. For SG/DP, the input to the heuristics is the optimal beginning time for each operation, and for SB/FZDP, the input is the beginning time associated with the largest weight for each operation. Both algorithms are stopped after the same amount of CPU time, and the results are summarized in Table 3. The testing shows significant performance improvement by using SB/FZDP, as it can efficiently provide good directions.

As mentioned early, the surrogate subgradient method combined with dynamic programming (SSG/DP) was developed recently, and with its interleaved idea, it generates better results than the subgradient methods or bundle methods for very large job shop scheduling problems (Ref. 3–4). In order to further improve SSG/DP, the interleaved idea is combined with FZDP, and the resulting SSG/FZDP is compared with SSG/DP. Twenty-five data sets were randomly generated with operation processing times, machine types required, and part due dates uniformly distributed within appropriate intervals. For each data set, there are 100 parts each with tardiness weight equal to 1, ten operations per part, and ten machine types in total. There are 10,000 multipliers in the dual problem. The duality gaps obtained by both methods are compared, and the improvement of the dual values in percentage by using SSG/FZDP is shown in Fig. 9 for all the 25 cases. It can be seen that the dual cost of SSG/FZDP is better than that of SSG/DP for most cases, and the average improvement is about 0.5%. In addition, the average duality gap is reduced by more than 10%. This much significant improvement in the duality gap is probably caused by the more sensible operation beginning times obtained by using fuzzy state weights in FZDP than the operation beginning times obtained by using crisp DP.

Table 3. Comparison of SB/FZDP via SG/DP.

Primal dimensions MT/P/O	Optimization method	Dual cost	Primal cost	Duality gap	CPU time
10/10/10	SB/FZDP	6668	6879	3.2%	30
	SG/DP	6234	7048	13.1%	30
10/40/10	SB/FZDP	37108	40395	8.9%	120
	SG/DP	35028	40395	15.3%	120
10/100/10	SB/FZDP	302025	341201	12.9%	600
	SG/DP	285942	341390	19.4%	600

CPU time in seconds. MT/P/O represents the number of machine types (MT), number of parts (P), and number of operations (O).

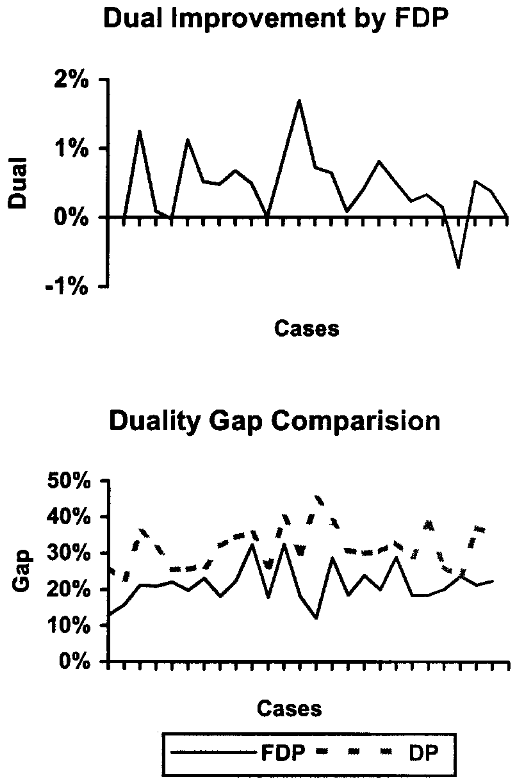

Figure 9. Testing results for SSG/FDP and SSG/DP.

7. Conclusions

Bundle methods are advanced methods developed for general nonsmooth optimization. When they are applied to optimize the dual functions within the Lagrangian relaxation framework, they can be improved and simplified, leading to the improved bundle method and the simplified bundle method with fuzzy dynamic programming as presented in this paper. These methods can utilize efficiently the valuable information contained in near-minimum solutions, which many times are byproducts when the relaxed problem is solved. Thus, they provide a new approach with good search directions and small computation requirements to solve large dual problems.

References

1. Geoffrion, M., *Lagrangian Relaxation for Integer Programming*, Mathematical Programming Studies, North Holland, Amsterdam, Netherlands, pp. 82–114, 1974.
2. Bertsekas, D. P., *Nonlinear Programming*, 2nd Edition, Athena Scientific, Belmont, Massachusetts, 1999.
3. Zhao, X., Luh, P. B., and Wang, J., *Surrogate Gradient Algorithm for Lagrangian Relaxation*, Journal of Optimization Theory and Applications, Vol. 100, pp. 699–712, 1999.
4. Kaskavelis, C. A., and Caramanis, M. C., *Efficient Lagrangian Relaxation Algorithms for Industry Size Job-Shop Scheduling Problems*, IIE Transactions, Vol. 30, pp. 1085–1097, 1998.
5. Hiriart-Urruty, J. B., and Lemarechal, C., *Convex Analysis and Minimization Algorithms*, Vols. 1 and 2, Springer Verlag, Berlin, Germany, 1993.
6. Nemhauser, G., and Wolsey, L., *Integer and Combinatorial Optimization*, John Wiley and Sons, New York, NY, 1988.
7. Held, M., Wolfe, P., and Crowder, H., *Validation of Subgradient Optimization*, Mathematical Programming, Vol. 6, pp. 62–88, 1974.
8. Kiwiel, K. C., *Restricted Step and Levenberg–Marquardt Techniques in Proximal Bundle Methods for Nonconvex Nondifferentiable Optimization*, SIAM Journal on Optimization, Vol. 6, pp. 227–249, 1996.
9. Kohl, N. and Madsen, O. B. G., *An Optimization Algorithm for the Vehicle Routing Problem with Time Windows Based on Lagrangian Relaxation*, Operations Research, Vol. 45, pp. 395–406, 1997.
10. Mifflin, R., Sun, D., and Qi, L., *Quasi-Newton Bundle-Type Methods for Nondifferentiable Convex Optimization*, SIAM Journal on Optimization, Vol. 8, pp. 583–603, 1998.
11. Wang, J., Luh, P. B., Zhao, X., and Wang, J., *An Optimization-Based Algorithm for Job Shop Scheduling*, Sadhana, Vol. 22, pp. 241–256, 1997.
12. Bellman, R., *Applied Dynamic Programming*, Princeton University Press, Princeton, New Jersey, 1957.
13. Kiwiel, K. C., *Proximal Level Bundle Methods for Convex Nondifferentiable Optimization, Saddle-Point Problems, and Variational Inequalities*, Mathematical Programming, Vol. 69, pp. 89–109, 1995.
14. Hoitomt, D. J., Luh, P. B., and Pattipati, K. R., *A Practical Approach to Job Shop Scheduling Problems*, IEEE Transactions on Robotics and Automation, Vol. 9, pp. 1–13, 1993.