This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

A Scalable Solution Methodology for Mixed-Integer Linear Programming Problems Arising in Automation

Mikhail A. Bragin, *Member, IEEE*, Peter B. Luh, *Life Fellow, IEEE*, Bing Yan, *Member, IEEE*, and Xiaorong Sun, *Student Member, IEEE*

***Abstract*— Many operation optimization problems such as scheduling and assignment of interest to the automation community are mixed-integer linear programming (MILP) problems. Because of their combinatorial nature, the effort required to obtain optimal solutions increases drastically as the problem size increases. Such operation optimization problems typically need to be solved several times a day and require short solving times (e.g., 5, 10, or 20 min). The goal is, therefore, to obtain near-optimal solutions with quantifiable quality in a computationally efficient manner. Existing MILP methods, however, suffer from slow convergence and may not efficiently achieve this goal. In this paper, motivated by fast convergence of augmented Lagrangian relaxation (LR), a novel advanced price-based decomposition and coordination "surrogate absolute-value LR" (SAVLR) approach is developed. Within the method, convergence of our recent surrogate LR (SLR), which has overcome all major difficulties of traditional LR, is significantly improved by penalizing constraint violations by adding "absolute-value" penalties. Moreover, such penalties are efficiently linearized in a standard way, thereby enabling the use of MILP solvers. By exploiting the beautiful property of exponential reduction of complexity of subproblems upon decomposition, subproblems are efficiently solved and their solutions are efficiently coordinated by updating Lagrangian multipliers. Convergence is then proved under novel adjustment of penalty coefficients. A series of generalized assignment problems is considered, and for these problems, superior performance of SAVLR over other state-of-the-art and state-of-the-practice methods is demonstrated. Accompanying CPLEX codes, whereby SAVLR is implemented, are also included.**

***Note to Practitioners*—Examples of important problems that arise in automation community include scheduling and assignment problems. Because of their combinatorial nature, the effort required to obtain optimal solutions increases drastically as the problem size increases. Existing mixed-integer linear programming (MILP) methods, however, may suffer from slow convergence and may not efficiently achieve this goal. The new method revolutionizes the way such problems can be solved with major improvements on the overall performance. It is based on our recent breakthrough "surrogate Lagrangian relaxation" (LR), which has overcome all major difficulties of traditional LR while exploiting the beautiful property of exponential reduction of complexity upon decomposition. To significantly improve convergence while maintaining linearity so as to use MILP solvers, our idea is to penalize violations of relaxed constraints by the infrequently used "absolute-value" penalty functions. Although not differentiable, absolute-value penalties have the advantage of being exactly linearizable through extra variables and constraints. The difficulties caused by those extra constraints, which couple subproblems, are resolved by adaptive adjustment of penalty coefficients. A series of generalized assignment problems is considered and superior performance of the new method against state-of-the-art and state-of-the-practice methods is demonstrated. Accompanying CPLEX codes whereby the new method is implemented are also included.**

***Index Terms*— Absolute-value penalty, branch-and-cut (B&C), generalized assignment problems (GAPs), linearity, mixed-integerlinear programming (MILP), separability, surrogate Lagrangian relaxation (SLR), surrogate absolute-value Lagrangian relaxation (SAVLR).**

I. Introduction

MANY practical systems of importance to the automation community such as manufacturing scheduling [1]–[8] and generalized assignment problems (GAPs) [9], [10] are created by establishing subsystems first and then by loosely coupling them together to form the overall system. Optimizing system performance is often formulated as mixed-integer linear programming (MILP) problems (with integer linear programming (ILP) problems being a special case). The corresponding objective functions are additive in terms of cost components associated with each subsystem. In addition, constraints that couple subsystems are linear, therefore, are also additive in terms of subsystems. Such MILP problems are thus always separable. The difficulty solving MILP problems, however, is caused by the presence of integer decision variables. These variables lead to combinatorial complexity, which becomes very high when the problem size is large. As a result, the effort required to obtain optimal solutions to MILP problems increase exponentially. Since operation optimization problems typically need to be solved several times a day and require short solving times (e.g., 5, 10, or 20 min), for practical purposes the goal is to obtain near-optimal solutions with quantifiable quality in a computationally efficient manner. However, this goal may not always be achieved by using

Manuscript received March 18, 2018; accepted May 6, 2018. This paper was recommended for publication by Associate Editor K.-H. Chang and Editor L. Shi upon evaluation of the reviewers' comments. This work was supported in part by the National Science Foundation under Grant ECCS-1509666 and by UConn Academic Plan Level 1 Award 2806600. *(Corresponding author: Mikhail A. Bragin.)*

The authors are with the Department of Electrical and Computer Engineering, University of Connecticut, Storrs, CT 06269-4157 USA (e-mail: mikhail.bragin@uconn.edu; peter.luh@uconn.edu; bing.yan@uconn.edu; xiaorong.sun@uconn.edu).

This paper has supplementary downloadable material available at http://ieeexplore.ieee.org. It contains CPLEX codes implementing the SAVLR approach (as well as SLR, ADMM and B&C) for Examples 1, 2 and 3 of this paper. The total size of the files is 2.42 MB.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TASE.2018.2835298

1545-5955 © 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

state-of-the-art and state-of-the-practice MILP optimization methods because of fundamental difficulties as explained ahead.

Standard MILP methods such as branch-and-cut (B&C) [11]–[20] exploit linearity. The idea of the method is to obtain the convex hull, the smallest convex set that contains feasible solutions, whereby feasible solutions (or "integer solutions" if the original problem is an ILP problem) are located at its vertices. In this case, the problem of solving an MILP problem reduces to solving a linear programming (LP) problem by using LP methods. To obtain the convex hull, B&C cuts off LP regions without cutting off feasible solutions by using linear "valid inequalities" (or cuts). If the convex hull is difficult to obtain, the method resorts to time-consuming branch and bound (B&B) and heuristics to obtain feasible solutions.

A traditional decomposition and coordination Lagrangian relaxation (LR) method [21]–[37] exploits separability into subproblems, each with much reduced complexity. However, because standard LR requires solving all subproblems to update multipliers, the relaxed problem is difficult to fully optimize and multipliers can suffer from severe zigzagging. Moreover, convergence proof and implementation require the knowledge of the optimal dual value. While adaptive estimates of the optimal dual value have been used to guarantee convergence, such adaptive adjustments may require many iterations. As a result of high computational effort, zigzagging, and inefficient estimations of the optimal dual value, the overall convergence of the method is slow.

In this paper, MILP optimization methods are reviewed in Section II. Major difficulties of standard LR have been overcome by our recent SLR [38], whereby proper "surrogate subgradient directions" are obtained by requiring the satisfaction of only the simple "surrogate optimality condition" [39]. When solving separable problems, the surrogate optimality condition is automatically satisfied after solving one or several subproblems at a time and resulting surrogate subgradient directions are smooth, thereby alleviating zigzagging and reducing computational requirements. However, the difficulty is that levels of constraint violations may not reduce sufficiently fast, thereby causing difficulties searching feasible solutions. Moreover, lower bounds generated by the method may not provide a sufficiently good measure of solutions quality. These difficulties have been demonstrated when solving power systems' unit commitment (UC) problems with a significant number of combined cycle (CC) units [40], [41].

In Section III, a GAP is introduced first to convey ideas by providing an example of a separable ILP problem, and then a general separable MILP problem formulation is presented.

In Section IV, computational difficulties associated with existing methods are overcome by developing a novel advanced price-based decomposition and coordination "surrogate absolute-value LR" (SAVLR). Within the method, convergence of our recent SLR is significantly improved by penalizing constraint violations using "absolute-value" penalties with exact linearization through extra variables and constraints. While these extra constraints couple subproblems and the surrogate optimality condition may not be satisfied when penalty coefficients are too large, convergence is guaranteed by a novel adjustment of penalty coefficients. Moreover, relaxed problems are decomposed into MILP subproblems with drastically reduced complexity, and their solutions are efficiently coordinated, thereby resulting in fast convergence.

In Section V, to demonstrate the performance of SAVLR and to compare with SLR, alternate direction method of multipliers (ADMM) and standard B&C, a series of GAPs is considered including problems with machine availability, job release, and sequence-dependent setup times. It is demonstrated that for these problems, SAVLR is much faster compared to other methods.

II. Literature Review

Existing MILP methods such as B&C and LR are used to obtain feasible solutions while quantifying their quality and their difficulties are reviewed in Section II-A. Recent surrogate subgradient method (SSM) and SLR that improve convergence of standard LR are reviewed in Section IV-B. Other methods such as genetic algorithm that does not provide a measure of solutions quality and heuristics are excluded. Also, the branch-and-bound (B&B) method, which is essentially as the "tail end" or "the last resort" of B&C, is mentioned only briefly.

A. Relevant Optimization Methods

When solving MILP problems, state-of-the-art and state-of-the-practice methods may not obtain near-optimal solutions with quantifiable quality in a computationally efficient manner because of fundamental difficulties as explained next.

1) Branch-and-Cut (B&C) [11]–[20]: When solving an LP problem, its convex hull (the smallest convex set containing all feasible solutions) is piecewise linear and is identical to the feasible set. An optimal solution is at one of its vertices. When solving an MILP problem, however, the feasible set of the integrality relaxed LP problem is typically larger than the convex hull, and the resulting LP solution is typically not feasible with respect to the original MILP problem. B&C attempts to obtain the convex hull by cutting off LP regions without cutting off feasible solutions using linear "valid inequalities" (or cuts). If the convex hull is obtained, feasible solutions (or "integer solutions" if an original problem is an ILP problem) are located its vertices. In this case, the problem of solving an MILP problem reduces to solving an LP problem by using LP methods. When the convex hull is difficult to obtain, the method relies on B&B and heuristics to obtain feasible solutions. When solving Midcontinent Independent System Operator's (the largest independent system operator in the USA) UC problems with a large number of CC units, cutting operations typically cannot obtain the convex hull, and a significant number of branching operations is then required to obtain feasible solutions. For these problems, we have vividly witnessed the breakdown of B&C. The fundamental difficulty, we believe, is that B&C has no "local" concept. Without exploiting "local" subsystem features, constraints associated with transitions among configurations within one CC unit are treated "globally," thereby affecting the solution process of the entire problem, thereby leading to long CPU times and large mixed-integer programming (MIP) gaps [40], [41].

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

2) Lagrangian Relaxation (LR): LR in combination with subgradient methods was traditionally used to solve MILP problems by exploiting separability [21]–[37]. After relaxing system-wide coupling constraints and decomposing the relaxed problem into subproblems associated with individual subsystems, subproblem solutions are coordinated by iteratively updating Lagrangian multipliers. The major difficulties of standard LR with subgradient methods are that: 1) they require solving all subproblems to update multipliers, leading to high computational effort and zigzagging of multipliers and 2) the convergence proof requires the knowledge of the optimal dual value, which is generally unknown. While convergence can be achieved in practice by adaptively estimating the optimal dual value, such process is typically inefficient. As a result of the above difficulties, overall convergence may be very slow.

3) Augmented Lagrangian Relaxation (ALR) and Alternate Direction Method of Multipliers (ADMM): Augmented LR (ALR) was first introduced in the late 1960s by Hestenes [42] and Powell [43], and is a powerful method to improve convergence of standard LR by penalizing constraint violations using quadratic penalties [44]–[46]. Under assumptions of convexity and smoothness of the objective function and constraints, convergence was proven. The major difficulty is that with the introduction of quadratic penalties, the overall problem is no longer linear nor separable. To reduce the effort in optimizing the relaxed problem, ADMM was introduced in 1970s [47]–[50]. Within ADMM [50, pp. 13–14], two subproblems are formulated by fixing selected decision variables. When problems are large, however, each subproblem may also be significant in size and complexity. Moreover, when solving MILP problems, the method typically does not converge because stepsizes within ADMM do not approach zero, which is required for convergence when optimizing associated nonsmooth dual functions [50, p. 73].

B. Recent Developments

1) Surrogate Subgradient Method (SSM) [38] and Surrogate Lagrangian Relaxation (SLR) [39]: Computational difficulties associated with standard LR have been overcome by SSM [39]. Within SSM, "surrogate subgradient directions" that form acute angles with directions toward the optimal multipliers are obtained after solving one subproblem at a time only subject to the simple "surrogate optimality condition" [39] with much reduced computational effort and zigzagging. As a result, by updating multipliers along these directions with appropriately chosen stepsizes, multipliers get closer to the optimal multipliers. However, within SSM [39], stepsizes require the knowledge of the optimal dual value for convergence proof as well as for practical implementations. Within our recent SLR [38], convergence has been proven without requiring the knowledge of the optimal dual value. This was achieved with a constructive process based on the contraction mapping concept, whereby distances between Lagrange multipliers at consecutive iterations decrease, and as a result, multipliers converge to a unique limit. At the same time, stepsizes are kept sufficiently large to avoid premature algorithm termination. In addition, a constructive stepsizing formula satisfying these criteria has been developed.

2) Combination of SLR and B&C [40], [41], [51]–[54]: SLR has been combined with B&C, whereby B&C has been used to solve subproblems. However, within the method, levels of constraint violations may not reduce fast enough. For MILP problems such as UC with a significant number of CC units that arise in power systems, computational improvements over standard B&C may not be significant enough.

III. MILP Problem Formulation

In Section III-A, the GAP [9], [55] is presented to convey ideas by providing an example of a "separable" ILP problem. In Section III-B, a general MILP problem formulation is presented, whereby a GAP is as a special case.

A. Motivating Generalized Assignment Problem

The goal of GAPs is to minimize the total assignment cost while assigning a set of jobs I to a set of machines J, while making sure that every job i is assigned to only one machine j, and that machines' available capacity T_j is not exceeded by assigned jobs [9], [55]. Assignment of a job i to a machine j is captured through assignment binary variables $y_{i,j}$, which are equal to 1 if and only if job i is assigned to a machine j. With each assignment, there is an associated cost $g_{i,j}$ and a time $t_{i,j}$ that a job i requires to be processed on a machine j. The problem is formulated in the following way:

$$\min_{y_{i,j}} \sum_{i=1}^{I}\sum_{j=1}^{J} g_{i,j}\, y_{i,j}, \quad y_{i,j} \in \{0,\ 1\},\ \ g_{i,j} \geq 0$$
$$t_{i,j} \geq 0, \quad T_j \geq 0 \tag{1}$$

$$\text{s.t.} \sum_{i=1}^{I} t_{i,j}\, y_{i,j} \leq T_j, \quad j = 1, \ldots, J \tag{2}$$

$$\sum_{j=1}^{J} y_{i,j} = 1, \quad i = 1, \ldots, I. \tag{3}$$

The problem (1)–(3) can be viewed as machine subsystems, which are subject to machine capacity constraints (2), coupled together by job assignment constraints (3). This problem is separable because the objective function (1) and system-wide coupling constraints (3) are additive in terms of machine subsystems. The problem is an ILP problem, a special case of MILP problems introduced next.

B. General Formulation of MILP Problems

As reviewed in introduction, practical systems of importance are created by establishing subsystems and by coupling them together to form the overall system. Optimizing system performance is formulated as MILP problems. The objective function of such problems is written in the following generic way:

$$\min_{x,y}\{(d^x)^T x + (d^y)^T y\}, (x, y) \in \Omega \subset R^{n_x} \times Z^{n_y} \tag{4}$$

where d^x and d^y are $n^x \times 1$ and $n^y \times 1$ cost column vectors and x and y are an $n^x \times 1$ and $n^y \times 1$ decision column vectors, consisting of pairwise disjoint decision column vectors

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

$\{(x_j, y_j)\}_{j=1,\ldots,J}$, where x_j and y_j are $n_j^x \times 1$ and $n_j^y \times 1$ column vectors such that $\sum_j n_j^x = n^x$ and $\sum_j n_j^y = n^y$. Individual subsystems are subject to subsystem constraints

$$A_j^x x_j + A_j^y y_j \le b_j, (x_j, y_j) \in \Omega_j, \quad j = 1, \ldots, J \tag{5}$$

where matrices A_j^x and A_j^y have dimensions $m_j \times n_j^x$ and $m_j \times n_j^y$, b_j are $m_j \times 1$ column vectors; Subsystems are coupled through system-wide coupling constraints

$$A^{x,0} x + A^{y,0} y = b^0. \tag{6}$$

Matrices $A^{x,0}$ and $A^{y,0}$ have dimensions $m_0 \times n^x$ and $m_0 \times n^y$, and b^0 is an $m_0 \times 1$ column vector. Because of linearity, objective function (4) and system-wide coupling constraints (6) are additive in terms of subsystems, therefore, problems (4)–(6) are always separable. Existence of solutions to (4)–(6) is ensured per following assumption.

Assumption 1: The feasible set of (4)–(6) is nonempty and domain Ω is bounded.

In terms of (1)–(3), $y = (y_{1,1}, \ldots, y_{i,j}, \ldots, y_{I,J})^T \in \Omega = \{0, 1\}^{I \cdot J}$, and d^T consists of assignment costs $g_{i,j}$. First J elements of the first row of A^0 are 1 and remaining are zero; first J elements of the second row of A^0 are 0, next J elements are 1, and the remaining are zeros. All I elements of b^0 are 1. Since there is only one constraint (2) per subsystem, each matrix A_j is a vector consisting of 0 and $t_{i,j}$, and each vector b_j is a scalar T_j.

IV. Solution Methodology

This section is to overcome difficulties of existing methods presented in Section II. Motivated by the idea of the method of multipliers (frequently referred to as ALR) [42]–[46], novel SAVLR is developed. Within SAVLR, our recent SLR is enhanced by using "absolute-value" penalty terms, thereby improving convergence and enabling the use of MILP solvers in Section IV-A. In Section IV-B, convergence of SAVLR is proven through novel adjustment of penalty coefficients to satisfy the "surrogate optimality condition," thereby guaranteeing overall convergence of the method. In Section IV-C, practical implementation of SAVLR is discussed.

A. Surrogate Absolute-Value Lagrangian Relaxation

To improve convergence of SLR [38], constraint violations of relaxed constraints are penalized by using "absolute-value" penalty terms with positive penalty coefficients $\{c^k\}$. The "absolute-value" relaxed problem is formulated as

$$\min_{x,y} L_{c^k}(x, y, \lambda^k), \quad \text{s.t., } (5), (x, y) \in \Omega, \ \lambda^k \in R^m \tag{7}$$

where

$$L_{c^k}(x, y, \lambda^k) \equiv (d^x)^T x + (d^y)^T y + (\lambda^k)^T g(x, y) + \frac{c^k}{2} \|g(x, y)\|_1 \tag{8}$$

is the "absolute-value" Lagrangian *function* with $||\cdot||_1$ denoting an L^1-norm, and $g(z)$ denoting levels of constraint violations

$$g(x, y) = A^{x,0} x + A^{y,0} y - b^0. \tag{9}$$

If some or all system-wide coupling constraints are inequalities, they are first converted into equalities by introducing nonnegative slack variables, and the "absolute-value" Lagrangian function is formed exactly as in (8).

Because the domain is bounded per Assumption 1, problem (7) has a bounded solution and a bounded value of $g(x, y)$ for any $(x, y) \in \Omega$ and for any positive c^k. Moreover, the SLR framework can be used to establish convergence because the function (8) can be viewed as a Lagrangian function for the following problem[1]:

$$\min_{x,y} \left\{ (d^x)^T x + (d^y)^T y + \frac{c^k}{2} \|g(x, y)\|_1 \right\} \quad \text{s.t. } (5), \quad (6), \ (x, y) \in \Omega. \tag{10}$$

Moreover, subproblems can be formed from (7) by selecting variables associated with one subsystem as decision variables and fixing variables associated with other subproblems at previously obtained values as

$$\min_{x_j, y_j} \ L_{c^k}\left(x_j, x_{-j}^{k-1}, y_j, y_{-j}^{k-1}, \lambda^k\right), \quad \text{s.t. } (5), (x_j, y_j) \in \Omega_j, \quad \lambda^k \in R^m. \tag{11}$$

For compactness of notation, x_{-j} and y_{-j} with subscripts "$-j$" mean that x_{-j} and y_{-j} are components of x and y without x_j and y_j, respectively. In case whereby there are slack variables, such variables are not fixed within subproblems. Also, subproblems can be formed by selecting variables associated with several subsystems and fixing decision variables associated with other subsystems.

Following standard practice,[2] subproblems are linearized exactly and subproblem j can be written as an MILP subproblem after introducing continuous decision variables q_i as

$$\min_{x_j, y_j, \{q_i\}_{i=1,\ldots,I}} \left\{ \begin{array}{l} \left(d_j^x\right)^T x_j + \left(d_j^y\right)^T y_j \\ +(\lambda^k)^T g\left(x_j, x_{-j}^{k-1}, y_j, y_{-j}^{k-1}\right) + \dfrac{c^k}{2} \displaystyle\sum_i q_i \end{array} \right\} \tag{12}$$

$$\text{s.t. } (5), \quad -q_i \le g_i\left(x_j, x_{-j}^{k-1}, y_j, y_{-j}^{k-1}\right) \le q_i \quad i = 1, \ldots, I. \tag{13}$$

Since the complexity of subproblems (12), (13) is much reduced upon the decomposition, obtaining subproblem solutions is much easier compared to that of the original problem (4)–(6).

Within SLR, one subproblem, which may consist of one or few subsystems, is solved at a time, and a solution (x_j^k, y_j^k) to subproblems (12), (13) should satisfy the

[1] Within SLR [38], linearity of the objective function is not required, but linearity of constraints [38, p. 178] and boundedness of constraint norms [38, p. 176] are required. Linearity of constraints is immediate from (6), and boundedness follows from linearity of (6) and Assumption 1.

[2] The linearization of absolute-value functions is performed in a standard way based upon [56, p. 28]. Consider a simple problem: $\min_{x,y}\{x + |y - a^y|\}$. This problem is linearized by introducing a continuous decision variable q^y, and two constraints. The linearized problem can then be equivalently written as

$$\min_{x,y,q^y} \ \{x + q^y\}, \quad \text{s.t. } -q^y \le y - a^y \le q^y.$$

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

surrogate optimality condition [38], [39]

$$\tilde{L}_{c^k}\left(x_j^k, x_{-j}^{k-1}, y_j^k, y_{-j}^{k-1}, \lambda^k\right) < \tilde{L}_{c^k}\left(x_j^{k-1}, x_{-j}^{k-1}, y_j^{k-1}, y_{-j}^{k-1}, \lambda^k\right). \quad (14)$$

Here

$$\tilde{L}_{c^k}\left(x_j^k, x_{-j}^{k-1}, y_j^k, y_{-j}^{k-1}, \lambda^k\right) \equiv \left(d_j^x\right)^T x_j^k + \left(d_j^y\right)^T y_j^k + (\lambda^k)^T \times g\left(x_j^k, x_{-j}^{k-1}, y_j^k, y_{-j}^{k-1}\right) + \frac{c^k}{2}\sum_i q_i^k \quad (15)$$

is a surrogate dual value defined for a solution (x_j^k, y_j^k) of a subproblem (12), (13) obtained at iteration k, and most recent solutions to other subproblems $(x_{-j}^{k-1}, y_{-j}^{k-1})$ obtained at previous iterations up to iteration $k-1$. Since within SLR only one of few subproblems are solved at a time, the surrogate dual value is higher than that of the dual value, which is obtained by solving all subproblems to optimality.

Because of integer decision variables involved in the optimization, the dual function is nonsmooth and polyhedral concave. As a result, a line search along subgradient directions may not lead to higher values of dual function. Nevertheless, subgradient directions always form acute angles with directions toward λ^*. Therefore, with appropriate stepsizes, it is possible to get closer to λ^*. Within SLR, to guarantee that surrogate subgradient directions defined as

$$\tilde{g}(x^k, y^k) = g(x, y)|_{x=x^k, y=y^k} \quad (16)$$

form acute angles with directions toward λ^*, a solution (x^k, y^k), which in (16) for brevity denotes all subproblem solutions obtained up to iterations k, needs to satisfy the surrogate optimality condition (14). Then, by updating multipliers

$$\lambda^{k+1} = \lambda^k + s^k \tilde{g}(x^k, y^k) \quad (17)$$

along these surrogate subgradient directions with appropriately chosen stepsizes, multipliers get closer to λ^* from one iteration to the next. Within SLR [38], [53], to guarantee convergence without requiring the optimal dual value, stepsizes are set as

$$s^k = \alpha_k \frac{s^{k-1}\|\tilde{g}(x^{k-1}, y^{k-1})\|_2}{\|\tilde{g}(x^k, y^k)\|_2}, \quad 0 < \alpha_k < 1 \quad (18)$$

where

$$\alpha_k = 1 - \frac{1}{Mk^\rho}, \quad \rho = 1 - \frac{1}{k^r}, \ M \geq 1, \ 0 < r < 1. \quad (19)$$

B. Convergence of Surrogate Absolute-Value Lagrangian Relaxation

As reviewed in Section II, when using SLR, the relaxed problem is decomposed into independent subproblems and the surrogate optimality condition is satisfied after solving one or few subproblems. However, within SAVLR, subproblem solutions are not independent as can be seen from constraints (13), which are introduced to linearize absolute-value penalty terms. Moreover, with very large penalties, constraint violations are forced to zero, thereby leading to a suboptimal feasible solution. As a result, the surrogate optimality condition may not be satisfied. In the following Proposition 1, it is established under which condition the surrogate optimality condition is satisfied.

Proposition 1: Satisfaction of the surrogate optimality condition: Within one iteration (after solving all subproblems exactly once) if $q_i^{k-1} \neq 0$ for at least one i, then the surrogate optimality condition is satisfied. Moreover, if $q_i^{k-1} = 0$ all i, then the surrogate optimality condition may not be satisfied.

Proof: If $q_i^{k-1} \neq 0$ then constraint violation is not zero for at least one system-wide constraint. Assume that there does not exist j, such that the surrogate optimality condition (14) is satisfied after solving a subproblem j. In this case, there exists no such value in (15) that satisfies (14). This is impossible because this would imply that there is no violation and the cost is optimal. There is a contradiction with the assumption that $q_i^{k-1} \neq 0$ for at least one i. When $q_i^{k-1} = 0$ all i, a feasible solution is found, which is generally not guaranteed to be optimal. □

For as long as the surrogate optimality condition is satisfied for at least one subproblem within one iteration, c^k is increased as

$$c^{k+1} = c^k \cdot \beta, \quad \beta > 1. \quad (20)$$

However, when c^k becomes too large, feasibility is overemphasized at the expense of optimality and from Proposition 1 it follows that the "surrogate optimality condition" may not be satisfied, thereby leading to suboptimal feasible solutions. At the other extreme, when c^k is zero, SAVLR becomes SLR, which has been proven to converge [38]. Therefore, the idea to satisfy the surrogate optimality condition is to reduce c^k whenever the surrogate optimality condition is not satisfied as

$$c^{k+1} = c^k/\beta, \quad \beta > 1. \quad (21)$$

In order not to increase the possibility that the surrogate optimality condition is violated again, c^k is no longer increased.

As proved in Proposition 1, when levels of constraint violations are zero, no subproblem solution will satisfy the surrogate optimality condition (14). It is also possible that when levels of constraint violations are close to zero, most of subproblem solutions will not satisfy (14). As a result, it may require solving many subproblems to satisfy (14). To speed up the process, c^k is reduced per (21) after a predetermined number of subproblems h $(<J)$ are solved without satisfying (14). In the following Theorem 1, convergence of SAVLR is proven.

Theorem 1 (Convergence of SAVLR): By updating multipliers (17) with stepsizes (18), (19) and adjusting penalty coefficients (20), (21), multipliers converge to λ_c^* that maximize the dual function

$$q_{c^k}(\lambda) \equiv \min_{x,y} L_{c^k}(x, y, \lambda). \quad (22)$$

Proof: There are two possible cases.

Case 1: During the entire iterative process, after a finite number of overall reductions of c^k, the surrogate optimality condition (14) is satisfied. Therefore, following Theorem 2.1 of SLR [38], SAVLR will converge.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

Case 2: After a sufficiently many iterations, condition (14) is still not satisfied. Then c^k approaches zero and SAVLR essentially becomes SLR. As proved in [38], SLR converges when multipliers are updated as in (17) and stepsizes are set as in (18) and (19). Therefore, SAVLR also converges. □

Corollary 1 *(Rate of Convergence):* SAVLR converges with a linear rate outside a sphere centered at λ_c^*. Moreover, the radius of this sphere is smaller than that for SLR.

Proof: As established in [38, p. 187, Proposition 2.5], assuming that there exists a scalar $\mu > 0$, multipliers approach λ^* with a linear rate of convergence outside of a sphere centered at λ^*. Moreover, the radius of the sphere is proportional to the norm of levels of constraint violations

$$\frac{s^k \|\tilde{g}_{\text{SLR}}(x^k, y^k)\|^2}{\mu} < \|\lambda^* - \lambda^k\|^2, \quad k = 0, 1, \ldots. \tag{23}$$

As established in Section III-A, SAVLR can be viewed as SLR with respect to a problem with the objective function (10). Therefore, within SAVLR, assuming that there exists a scalar $\mu_c > 0$, the following inequality also holds:

$$\frac{s^k \|\tilde{g}_{\text{SAVLR}}(x^k, y^k)\|^2}{\mu_c} < \left\|\lambda_c^* - \lambda^k\right\|^2, \quad k = 0, 1, \ldots. \tag{24}$$

Assuming μ and μ_c exist and are the same in value, then with much reduced constraint violations, multipliers within SAVLR get much closer to λ_c^* and with linear rate as compared to how close multipliers approach λ^* with linear rate within SLR. □

C. Implementation of the SAVLR Method

Key steps of the Algorithm. The key steps are as follows.

Step 0: Initialize λ^0, x^0, y^0, s^0, and c^0;

Step 1: Update α_k and s^k per (18) and (19), update λ^{k+1} per (17), and update c^{k+1} per (20);

Step 2: For given λ^{k+1}, solve subproblem (12) and (13). If the surrogate optimality condition (14) is satisfied, go to Step 4;

Step 3: If (14) is not satisfied, go to Step 2 and solve the next subproblem. If (14) is not satisfied after solving each subproblem, reduce the penalty coefficient c^k per (21) and go to Step 2;

Step 4: Check stopping criteria such as the CPU time, number of iterations, and surrogate subgradient norm. If satisfied, go to Step 5. Otherwise, go to Step 1;

Step 5: Search for feasible solutions. If a solution is found, go to Step 6. Otherwise, go to Step 1;

Step 6: Check duality gap. A duality gap can be calculated by using the best available feasible cost and the largest available dual value. If duality gap is satisfactory, then Stop. Otherwise go to Step 1.

1) Obtaining of Feasible Solutions: The process of obtaining feasible solutions is generally problem dependent since each problem may have its own structures. Solutions to subproblems are typically feasible with respect to subproblems, but these solutions may not satisfy relaxed constraints. To satisfy these constraints, some or all integer decision variables are fixed within the original problem (4)–(6) at y_i^k, the most recent values obtained by solving subproblems, and the resulting problem is solved by B&C. Because constraint violations are penalized and become much smaller than those for SLR, the effort spent on heuristics to obtain feasible solutions to original problems is much reduced. If a solution feasible with respect to the original problem is not obtained, multipliers are adjusted for a few more iterations, and a feasible solution is searched again.

2) General Applicability for MILP Problems: The method can still be used for problems without the separable structures considered in this paper. For these problems, however, many more constraints need to be relaxed and penalized, and the method may not be as effective as for problems with separable structures.

V. Numerical Testing

The SAVLR method is implemented in CPLEX 12.7.1, and is tested on a laptop with the processor Intel Xeon CPU E3-1535M v6 at 3.1-GHz, 32 GB of RAM, and Windows 10. To demonstrate the efficiency of SAVLR, a series of GAPs are considered, and SAVLR is tested against other methods: B&C, SLR, and ADMM [47, pp. 13–14].[3] In Example 1, standard GAPs are considered. In Example 2, machine availability and job release features are included. In Example 3, sequence-dependent setup time feature is added. A brief guide of how to obtain results and how to by running associated CPLEX codes are explained in "readme" files within each of the supplementary zip files as well as within each code.

Example 1 *(Generalized Assignment Problems):* As explained in Section III-A, the goal of the GAP is to minimize the total assignment cost while satisfying machine capacity and assignment constraints. Within SAVLR, a subproblem j associated with machine j is formulated as

$$\min_{y_{i,j}} \left\{ \sum_{i=1}^{I} g_{i,j} y_{i,j} + \lambda_i^k y_{i,j} + \frac{c^k}{2} q_i \right\} \tag{25}$$

$$\text{s.t. } (2), -q_i \le y_{i,j} + \sum_{l=1 \text{ to } J,\ l \ne j} y_{i,l}^{k-1} - 1 \le q_i \quad i = 1, \ldots, I. \tag{26}$$

In the OR-Library, there are five standard types of GAPs (A, B, C, D, and E) [9], [55], [57]. According to testing experience of [9], [55], problem instances type D are most difficult. To test performance of SAVLR, a problem instance type D with 20 machines and 1600 jobs (d201600) is considered first. SAVLR is very fast because of: 1) drastically reduced complexity of machine subproblems upon decomposition allowing solving such subproblems to optimality; 2) efficient coordination of subproblem solutions inherited from SLR; and 3) accelerated convergence accomplished through absolute-value penalties. As shown in Fig. 1, a feasible cost 97 828 is obtained after 1371 s, which to the best of our knowledge is better than the smallest value 97 837 reported in [9] and [55].

[3]Because of a significant, sometimes even prohibitive, the computational effort required to optimize the relaxed problem within the Augmented Lagrangian Relaxation (ALR) method, its decomposable version, referred to as ADMM is tested.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

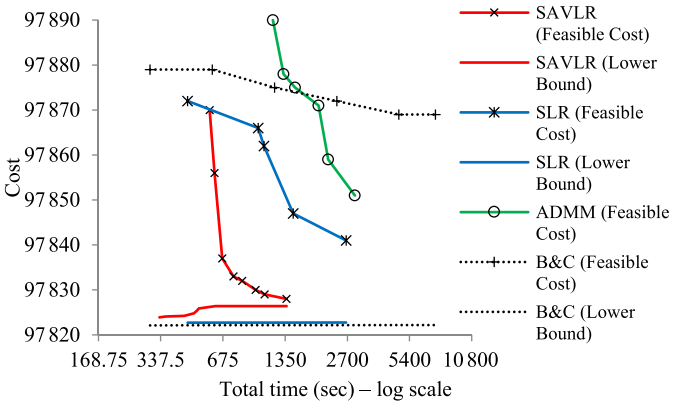

Fig. 1. Comparison of SAVLR versus SLR, ADMM, and B&C for the problem d201600 with 20 machines and 1600 jobs.

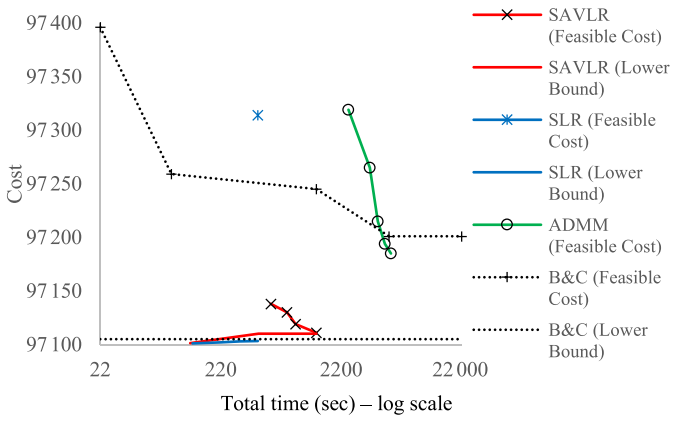

Fig. 2. Comparison of SAVLR versus SLR, ADMM, and B&C for the problem d401600 with 40 machines and 1600 jobs.

Within SLR, constant violations are not penalized and do not reduce fast. As a result, a feasible cost 97 841 is obtained after 2661 s. Within ADMM, two subproblems are formed by grouping every 10 machine subsystems. Subproblems created within ADMM are nonlinear and are much higher in complexity as compared to those created within SAVLR, thereby leading to very long, sometimes even prohibitive, solving times. To speed up the process, ADMM subproblems are solved with a 0.25% gap tolerance. Since subproblems are not solved to optimality, the lower bound for ADMM is not obtained and thus not reported. A feasible cost 97 851 is obtained after 2938 s. Within B&C, a feasible cost 97 869 is obtained after 4800 s.

A problem instance type D with 40 machines and 1600 jobs is considered next. As shown in Fig. 2, within SAVLR, the feasible cost 97 111 is obtained after 1183 s. Within SLR, the norm of constraint violations decreases much slower than that within SAVLR and only one feasible solution with the feasible cost 97 314 is found after 447 s without further improvement. Within ADMM, two subproblems are formed by grouping every 20 machine subsystems, and a feasible cost 97 185 is obtained after 5700 s. Within B&C, a feasible cost 97 201 is obtained after 5500 s.

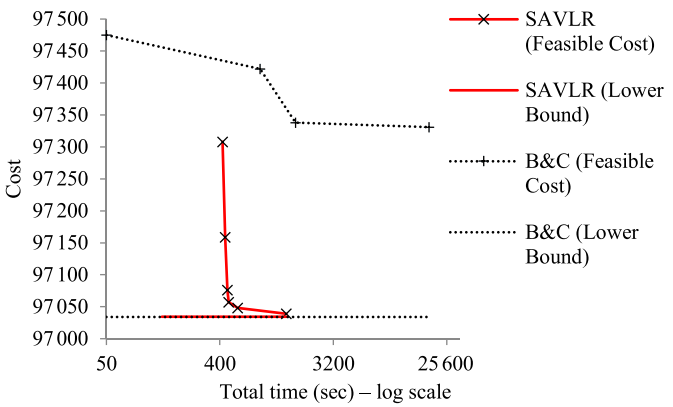

Fig. 3. Comparison of SAVLR versus B&C for the problem d801600 with 80 machines and 1600 jobs.

TABLE I
SCALABILITY RESULTS FOR EXAMPLE 1

Instance	Feasible Cost	CPU time (sec)
d201600	97 833	759
d401600	97 114	925
d801600	97 039	1350

Last, the largest instance of category D from the OR-Library [57] is considered with 80 machines and 1600 jobs (d801600), and results are shown in Fig. 3.

As shown in Fig. 3, SAVLR converges fast, thereby leading to good feasible solutions and tight lower bounds. Best feasible costs obtained within SAVLR are 97 048 after 554 s and 97 039 after 1350 s, and both to the best of our knowledge are better as compared to 97 052, the smallest value is reported in [9] and [55]. Within SLR, the norm of constraint violations decreases very slowly, and within ADMM the computational effort required to solve subproblems is very high. The results for these methods are not provided.

A. Scalability Results

To test scalability of SAVLR, three instances of GAPs are considered: d201600, d401600, and d801600. The stopping criterion for these instances is 0.01%, and the results are shown in Table I.

B. Comparison and Robustness of SAVLR Versus That of Branch-and-Cut

To test robustness, 30 simulations are performed after slightly perturbing parameters T_j for the problem instance with 20 machines and 1600 jobs (d201600).

Within SAVLR, the stopping criterion is 0.05% duality gap, and results are shown in Fig. 4 (red). All instances are solved within 300 s, and the average solving time is 210 s. As demonstrated in Fig. 4 (gray), B&C solves six instances within 500 s under the stopping criterion of 0.05% MIP gap. For the remaining 24 instances, it takes at least 1500 s.

Example 2 *(Generalized Assignment Problems With Machine Availability and Job Release Times):* Generally, a machine may not be available at the beginning of a period

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

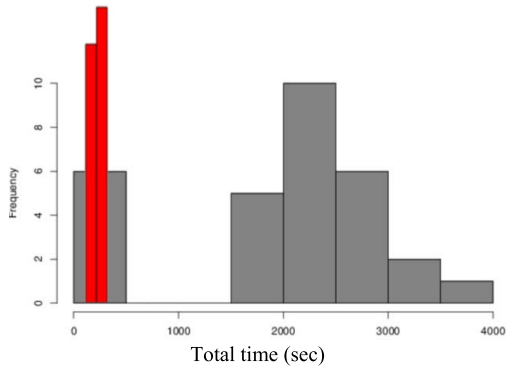

Fig. 4. Histogram showing performance of SAVLR (red) and B&C (gray) for solving problems with 20 machines and 1600 jobs.

until certain time a_j^{machine} and jobs may not be released until certain time a_i^{job}. To account for machine availability and job release, integer decision variables $b_{i,j}$ that denote the beginning of the processing time of job i on machine j are introduced that satisfy the following inequalities:

$$b_{i,j} \geq a_i^{\text{job}}, b_{i,j} \geq a_j^{\text{machine}}, \quad i = 1, \ldots, I, \ \ j = 1, \ldots, J. \tag{27}$$

Also, integer variables $c_{i,j}$ that denote job i completion time on machine j are introduced to ensure that all jobs are completed before the due date T_j

$$c_{i,j} \leq T_j, \quad i = 1, \ldots, I, \ \ j = 1, \ldots, J. \tag{28}$$

Moreover, completion time should be equal to the beginning time plus processing time if a job is assigned

$$y_{i,j} = 1 \Rightarrow c_{i,j} - b_{i,j} = t_{i,j}, \quad i = 1, \ldots, I, \ \ j = 1, \ldots, J. \tag{29}$$

Constraint (29) can be linearized using a "big-M" inequality as

$$t_{i,j} - M(1 - y_{i,j}) \leq c_{i,j} - b_{i,j} \leq t_{i,j} + M(1 - y_{i,j}), \quad i = 1, \ldots, I, \ \ j = 1, \ldots, J. \tag{30}$$

Lastly, time slots $[b_{i,j}, c_{i,j}]$ during which jobs are processed should not intersect, and this can be ensured by requiring that job i is either completed before any other job i' or job i is processed after any other job i' is completed

$$c_{i,j} \leq b_{i',j} \text{ OR } c_{i',j} \leq b_{i,j} \quad i \neq i', \ \ i, i' = 1, \ldots, I, \ \ j = 1, \ldots, J. \tag{31}$$

This logical inequality can be linearized after introducing extra binary variables as

$$c_{i,j} \leq b_{i',j} + M z^1_{i,i',j}; \ c_{i',j} \leq b_{i,j} + M z^2_{i,i',j}; \ z^1_{i,i',j} + z^2_{i,i',j} = 1, \quad i \neq i', \quad i, i' = 1, \ldots, I, \ \ j = 1, \ldots, J. \tag{32}$$

The objective function, machine capacity, and assignment constraints are the same as in (1)–(3).

GAP instances with 40 machines with a_j^{machine} and a_i^{job} are generated using uniform distributions U[0, 100] and U[0, 300], tested using SAVLR, SLR, ADMM, and B&C, and the results are shown in Figs. 5–7. To speed up computations, ADMM subproblems are solved with a 10% gap tolerance. Since subproblems are not solved to optimality, the lower bound for ADMM is not obtained and thus not reported.

SAVLR is significantly much more efficient compared to SLR, ADMM, and B&C because of the exponential reduction of complexity upon the decomposition, and efficient coordination of multipliers with accelerated convergence through penalization of constraint violations through novel "absolute-value" penalties with their exact linearization.

Example 3 *(Generalized Assignment Problems With Sequence-Dependent Setup Times):* Sequence-dependent setup times frequently arise because setup times to process jobs generally depend on the previous job and on the machine. This feature brings another layer of difficulty because the number of possible sequences of jobs grows fast as the number of jobs increases. In this example, sequences of jobs are captured through binary variables $x_{i,i',j}$ that take values of 1 if job i is assigned on machine j right before i' and 0 otherwise. Therefore, at most one job can be assigned right before job i'

$$\sum_{j=1}^{J} \sum_{i=1}^{I} x_{i,i',j} \leq 1, \quad i' = 1, \ldots, I \tag{33}$$

and at most one job can be assigned right before job i

$$\sum_{j=1}^{J} \sum_{i'=1}^{I} x_{i,i',j} \leq 1, \quad i = 1, \ldots, I. \tag{34}$$

A sequence of jobs (i, i') can only occur on at most one machine

$$\sum_{j=1}^{J} x_{i,i',j} \leq 1, \quad i, i' = 1, \ldots, I. \tag{35}$$

Also, a sequence of jobs (i, i') is assigned to machine j ($x_{i,i',j} = 1$) if both jobs i and i' are assigned to machine j ($y_{i,j} = 1$ and $y_{i',j} = 1$), and this condition can be formulated as

$$1 - x_{i,i',j} \leq M\big(2 - y_{i',j} - y_{i,j}\big), \quad i, i' = 1, \ldots, I, \quad j = 1, \ldots, J. \tag{36}$$

To avoid the simultaneous assignment of job i before job i' and assignment of job i' before job i, the following logical constraint is introduced:

$$1 - x_{i,i',j} \leq M\big(2 - y_{i',j} - y_{i,j}\big) \text{ OR } 1 - x_{i',i,j} \leq M\big(2 - y_{i',j} - y_{i,j}\big), \quad i, i' = 1, \ldots, I, \ \ j = 1, \ldots, J. \tag{37}$$

Constraint (37) can be linearized as explained in (32). The objective function (1) is modified to include setup costs as

$$\min_{x_{i,i',j}} \left\{ \sum_{i=1}^{I} \sum_{j=1}^{J} g_{i,j} y_{i,j} + \sum_{i=1}^{I} \sum_{i'=1}^{I} \sum_{j=1}^{J} h_{i,i',j} x_{i,i',j} \right\} \quad x_{i,i',j}, y_{i,j} \in \{0, 1\}, g_{i,j} \geq 0, \quad h_{i,i',j} \geq 0 \quad i, i' = 1, \ldots, I, \quad j = 1, \ldots, J. \tag{38}$$

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

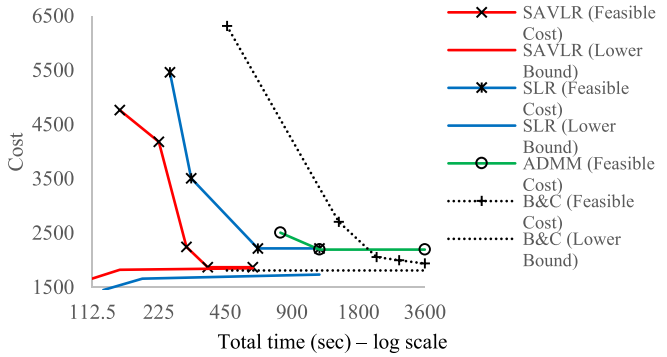

Fig. 5. Comparison of SAVLR versus SLR, ADMM, and B&C for the problem with 40 machines and 100 jobs and with a_j^{machine} and a_i^{job} generated using uniform distribution U[0, 100].

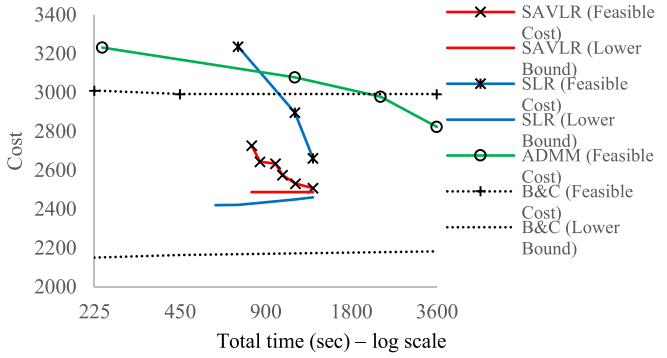

Fig. 6. Comparison of SAVLR versus SLR, ADMM, and B&C for the problem with 40 machines and 100 jobs and with a_j^{machine} and a_i^{job} generated using uniform distributions U[0, 100] and U[0, 300], respectively.

where $h_{i,i',j}$ is a setup cost of assigning job i is assigned on machine j right before i'. In a similar fashion, capacity constraints (2) can be modified as

$$\sum_{i=1}^{I} t_{i,j}\, y_{i,j} + \sum_{i=1}^{I}\sum_{i'=1}^{I} s_{i,i',j}\, x_{i,i',j} \le T_j, \quad j = 1, \ldots, J \qquad (39)$$

where $s_{i,i',j}$ is time required to set up job i' after job i on machine j. The assignment constraint (3) remains unchanged.

To demonstrate performance of SAVLR, the problem instance with 40 machines and 100 jobs is considered. Costs $h_{i,i',j}$ and setup times $s_{i,i',j}$ are generated using uniform distributions U[0, 100] and U[0, 20], respectively. Results as well as comparison against SLR and B&C are shown in Fig. 8. Comparison against ADMM is not included because of the very high computational effort required to solve subproblems.

As shown in Fig. 8, within SAVLR, the overall performance is much better as compared to that of SLR and standard B&C.

C. Comparison and Robustness of SAVLR Versus Those of Branch-and-Cut

To test robustness, 100 simulations are performed after slightly perturbing T_j. The stopping criterion is 10 min for SAVLR and 60 min for B&C.

As shown in Fig. 9, SAVLR is more robust and much more efficient: the duality gap obtained by SAVLR after 10 min

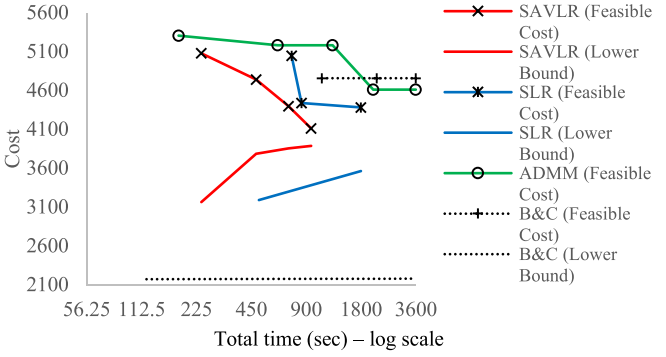

Fig. 7. Comparison of SAVLR versus SLR, ADMM, and B&C for the problem with 40 machines and 100 jobs and with a_j^{machine} and a_i^{job} generated using uniform distribution U[0, 300].

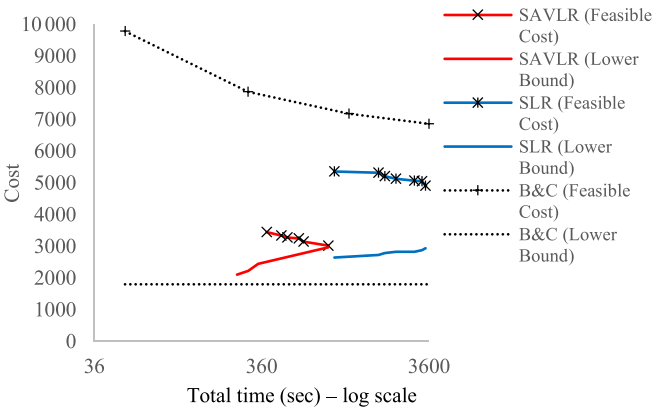

Fig. 8. Comparison of SAVLR and standard B&C for GAP with sequence-dependent setup times with 40 machines and 100 jobs.

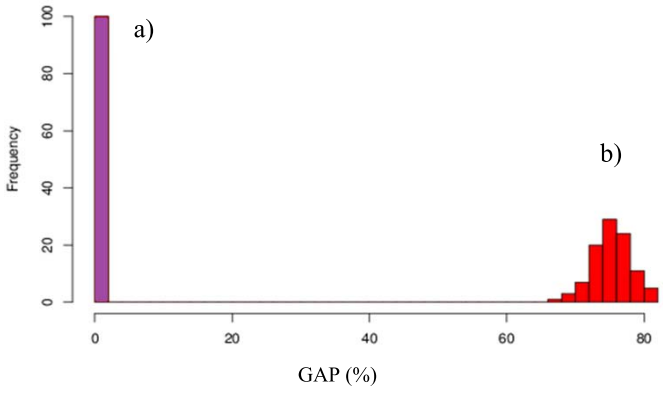

Fig. 9. Histogram showing performance of (a) SAVLR and (b) B&C for problems with 40 machines and 100 jobs.

is less than 2% for all 100 simulations with the average gap 0.67%, which is drastically smaller as compared to MIP gaps within standard B&C even after 1 h of CPU time.

VI. Conclusion

In this paper, a novel advanced price-based decomposition and coordination SAVLR approach is developed. Within the method, convergence of our recent SLR, which overcame all major difficulties of standard LR, is significantly improved

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

by penalizing constraint violations by using "absolute-value" penalties, which have the advantage of being exactly linearized. Testing results demonstrate that SAVLR obtains much better solutions and converges much faster as compared to other methods. With such effective coordination of subproblem solutions, our capabilities to solve difficult MILP problems that arise in the automation community and beyond will be advanced in a major way.

REFERENCES

[1] A. Che, W. Lei, J. Feng, and C. Chu, "An improved mixed integer programming approach for multi-hoist cyclic scheduling problem," *IEEE Trans. Autom. Sci. Eng.*, vol. 11, no. 1, pp. 302–309, Jan. 2014.

[2] W.-C. Lee, "Single-machine scheduling with past-sequence-dependent setup times and general effects of deterioration and learning," *Optim. Lett.*, vol. 8, no. 1, pp. 135–144, 2014, doi: 10.1007/s11590-012-0481-9.

[3] S. Hatami, R. Ruiz, and C. Andrés-Romano, "Heuristics and metaheuristics for the distributed assembly permutation flowshop scheduling problem with sequence dependent setup times," *Int. J. Prod. Econ.*, vol. 169, pp. 76–88, Nov. 2015.

[4] A. Subramanian, M. Battarra, and C. N. Potts, "An iterated local search heuristic for the single machine total weighted tardiness scheduling problem with sequence-dependent setup times," *Int. J. Prod. Res.*, vol. 52, no. 9, pp. 2729–2742, 2014.

[5] N. Löhndorf, M. Riel, and S. Minner, "Simulation optimization for the stochastic economic lot scheduling problem with sequence-dependent setup times," *Int. J. Prod. Econ.*, vol. 157, pp. 170–176, Nov. 2014.

[6] S.-W. Lin and K.-C. Ying, "ABC-based manufacturing scheduling for unrelated parallel machines with machine-dependent and job sequence-dependent setup times," *Comput. Oper. Res.*, vol. 51, pp. 172–181, Nov. 2014.

[7] T. Eren and E. Güner, "A bicriteria scheduling with sequence-dependent setup times," *Appl. Math. Comput.*, vol. 179, no. 1, pp. 378–385, 2006.

[8] G. M. Komaki and V. Kayvanfar, "Grey Wolf Optimizer algorithm for the two-stage assembly flow shop scheduling problem with release time," *J. Comput. Sci.*, vol. 8, pp. 109–120, May 2015.

[9] M. Posta, J. A. Ferland, and P. Michelon, "An exact method with variable fixing for solving the generalized assignment problem," *Comput. Optim. Appl.*, vol. 52, no. 3, pp. 629–644, 2012.

[10] M. L. Fisher, "The Lagrangian relaxation method for solving integer programming problems," *Manage. Sci.*, vol. 27, no. 1, pp. 1–18, 1981.

[11] M. Padberg and G. Rinaldi, "A branch-and-cut algorithm for the resolution of large-scale symmetric traveling salesman problems," *SIAM Rev.*, vol. 33, no. 1, pp. 60–100, 1991.

[12] K. L. Hoffman and M. Padberg, "Solving airline crew scheduling problems by branch-and-cut," *Manage. Sci.*, vol. 39, no. 6, pp. 657–682, 1993.

[13] S. Ceria, G. Cornuéjols, and M. Dawande, "Combining and strengthening Gomory cuts," in *Proc. Int. Conf. Integer Program. Combinat. Optim.*, Copenhagen, Denmark: Springer-Verlag, 1995, pp. 438–451.

[14] A. Caprara and M. Fischetti, "0, 1/2-Chvátal–Gomory cuts," *Math. Program.*, vol. 74, no. 3, pp. 221–235, 1996.

[15] F. Eisenbrand, "Gomory-Chvátal cutting planes and the elementary closure of polyhedra," Ph.D. dissertation, Dept. Natural Sci. Technol., Univ. Saarlandes, Saarbrücken, Germany, 2000. [Online]. Available: http://www2.math.uni-paderborn.de/fileadmin/Mathematik/AG-Eisenbrand/publications/diss.pdf

[16] I. Aliev and A. N. Letchford. (2013). "Iterated Chvátal–Gomory cuts and the geometry of numbers." [Online]. Available: https://arxiv.org/abs/1306.6031

[17] R. E. Gomory, "An algorithm for the mixed integer problem," RAND Corp., Santa Monica, CA, USA, Tech. Rep. RAND-P-1885, 1960.

[18] R. E. Gomory, "Solving linear programming problems in integers," *Combinat. Anal.*, vol. 10, pp. 211–215, Jul. 1960.

[19] A. H. Land and A. G. Doig, "An automatic method of solving discrete programming problems," *Econometrica*, vol. 28, no. 3, pp. 497–520, Jul. 1960.

[20] J. E. Mitchell, "Branch-and-cut," in *Wiley Encyclopedia of Operations Research and Management Science*. Hoboken, NJ, USA: Wiley, 2010.

[21] X. Guan, P. B. Luh, H. Yan, and J. A. Amalfi, "An optimization-based method for unit commitment," *Int. J. Elect. Power Energy Syst.*, vol. 14, no. 1, pp. 9–17, 1992.

[22] X. Guan, P. B. Luh, H. Yen, and P. Rogan, "Optimization-based scheduling of hydrothermal power systems with pumped-storage units," *IEEE Trans. Power Syst.*, vol. 9, no. 2, pp. 1023–1031, May 1994.

[23] F. Zhuang and F. D. Galiana, "Towards a more rigorous and practical unit commitment by Lagrangian relaxation," *IEEE Trans. Power Syst.*, vol. TPWRS-3, no. 2, pp. 763–773, May 1988.

[24] Q. Zhai, X. Guan, and J. Cui, "Unit commitment with identical units successive subproblem solving method based on Lagrangian relaxation," *IEEE Trans. Power Syst.*, vol. 17, no. 4, pp. 1250–1257, Nov. 2002.

[25] W. Ongsakul and N. Petcharaks, "Unit commitment by enhanced adaptive Lagrangian relaxation," *IEEE Trans. Power Syst.*, vol. 19, no. 1, pp. 620–628, Feb. 2004.

[26] N. J. Redondo and A. J. Conejo, "Short-term hydro-thermal coordination by Lagrangian relaxation: Solution of the dual problem," *IEEE Trans. Power Syst.*, vol. 14, no. 1, pp. 89–95, Feb. 1999.

[27] S. Virmani, E. C. Adrian, K. Imhof, and S. Mukherjee, "Implementation of a Lagrangian relaxation based unit commitment problem," *IEEE Trans. Power Syst.*, vol. 4, no. 4, pp. 1373–1380, Nov. 1989.

[28] Y. M. Ermol'ev, "Methods of solution of nonlinear extremal problems," *Cybernetics*, vol. 2, no. 4, pp. 1–14, 1966.

[29] B. T. Polyak, "A general method for solving extremal problems," *Sov. Math. Doklady*, vol. 8, no. 3, pp. 593–597, 1967.

[30] N. Z. Shor, "The rate of convergence of the generalized gradient descent method," *Cybernetics*, vol. 4, no. 3, pp. 79–80, 1968.

[31] N. Z. Shor, "Generalized gradient methods for non-smooth functions and their applications to mathematical programming problems," (in Russian), *Econ. Math. Methods*, vol. 12, no. 2, pp. 337–356, 1976.

[32] U. Brännlund, P. O. Lindberg, A. Nõu, and J. E. Nilsson, "Railway timetabling using Lagrangian relaxation," *Transp. Sci.*, vol. 32, no. 4, pp. 358–369, 1998.

[33] M. L. Fisher, "An applications oriented guide to Lagrangian relaxation," *Interfaces*, vol. 15, no. 2, pp. 10–21, 1985.

[34] H. D. Sherali and G. Choi, "Recovery of primal solutions when using subgradient optimization methods to solve Lagrangian duals of linear programs," *Oper. Res. Lett.*, vol. 19, no. 3, pp. 105–113, 1996.

[35] J. M. Mulvey and H. P. Crowder, "Cluster analysis: An application of Lagrangian relaxation," *Manage. Sci.*, vol. 25, no. 4, pp. 329–340, 1979.

[36] J. Xu and R. Nagi, "Solving assembly scheduling problems with tree-structure precedence constraints: A Lagrangian relaxation approach," *IEEE Trans. Autom. Sci. Eng.*, vol. 10, no. 3, pp. 757–771, Jul. 2013, doi: 10.1109/TASE.2013.2259816.

[37] J. Tang, C. Yan, X. Wang, and C. Zeng, "Using Lagrangian relaxation decomposition with heuristic to integrate the decisions of cell formation and parts scheduling considering intercell moves," *IEEE Trans. Autom. Sci. Eng.*, vol. 11, no. 4, pp. 1110–1121, Oct. 2014.

[38] M. A. Bragin, P. B. Luh, J. H. Yan, N. Yu, and G. A. Stern, "Convergence of the surrogate Lagrangian relaxation method," *J. Optim. Theory Appl.*, vol. 164, no. 1, pp. 173–201, 2015.

[39] X. Zhao, P. B. Luh, and J. Wang, "Surrogate gradient algorithm for Lagrangian relaxation," *J. Optim. Theory Appl.*, vol. 100, no. 3, pp. 699–712, 1999.

[40] X. Sun, P. B. Luh, M. A. Bragin, Y. Chen, J. Wan, and F. Wang, "A decomposition and coordination approach for large-scale security constrained unit commitment problems with combined cycle units," in *Proc. IEEE Power Energy Soc. Gen. Meeting*, Jul. 2017, pp. 1–5.

[41] X. Sun, P. B. Luh, M. A. Bragin, Y. Chen, J. Wan, and F. Wang, "A novel decomposition and coordination approach for large day-ahead unit commitment with combined cycle units," *IEEE Trans. Power Syst.*, to be published, doi: 10.1109/TPWRS.2018.2808272.

[42] M. R. Hestenes, "Multiplier and gradient methods," *J. Optim. Theory Appl.*, vol. 4, no. 5, pp. 303–320, 1969.

[43] M. J. D. Powell, "A method for nonlinear constraints in minimization problems," in *Optimization*, R. Fletcher, Ed. New York, NY, USA: Academic, 1969.

[44] D. P. Bertsekas, *Nonlinear Programming*, 3rd ed. Belmont, MA, USA: Athena Scientific, 2016.

[45] D. Gabay and B. Mercier, "A dual algorithm for the solution of nonlinear variational problems via finite element approximation," *Comput. Math. Appl.*, vol. 2, no. 1, pp. 17–40, 1976.

[46] R. Glowinski and A. Marrocco, "Sur l'approximation, par elements finis d'ordre un, et la resolution, par penalisation-dualite, d'une classe de problems de Dirichlet non lineares," *Revue Francaise d'Automatique, Informatique, Recherche Operationelle*, vol. 9, pp. 41–76, 1975.

[47] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, Jan. 2011.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

[48] M. J. Feizollahi, S. Ahmed, and A. Sun, "Exact augmented Lagrangian duality for mixed integer linear programming," *Math. Program.*, vol. 161, nos. 1–2, pp. 365–387, 2017.
[49] Y. Ouyang, Y. Chen, G. Lan, and E. Pasiliao, Jr., "An accelerated linearized alternating direction method of multipliers," *SIAM J. Imag. Sci.*, vol. 8, no. 1, pp. 644–681, 2015.
[50] J. Yang and X. Yuan, "Linearized augmented Lagrangian and alternating direction methods for nuclear norm minimization," *Math. Comput.*, vol. 82, no. 281, pp. 301–329, 2012.
[51] M. A. Bragin, P. B. Luh, J. H. Yan, and G. A. Stern, "Surrogate Lagrangian relaxation and branch-and-cut for unit commitment with combined cycle units," in *Proc. IEEE Power Energy Soc. Gen. Meeting*, Jul. 2014, pp. 1–5.
[52] M. A. Bragin, P. B. Luh, J. H. Yan, and G. A. Stern, "Novel exploitation of convex hull invariance for solving unit commitment by using surrogate Lagrangian relaxation and branch-and-cut," in *Proc. IEEE Power Energy Soc. Gen. Meeting*, Jul. 2015, pp. 1–5.
[53] M. A. Bragin, P. B. Luh, J. H. Yan, and G. A. Stern, "An efficient approach for solving mixed-integer programming problems under the monotonic condition," *J. Control Decis.*, vol. 3, no. 11, pp. 44–67, 2016.
[54] B. Yan *et al.*, "Grid integration of wind generation considering remote wind farms: Hybrid Markovian and interval unit commitment," *IEEE/CAA J. Automatica Sinica*, vol. 4, no. 2, pp. 205–215, Apr. 2017.
[55] P. Avella, M. Boccia, and I. Vasilyev, "A computational study of exact knapsack separation for the generalized assignment problem," *Comput. Optim. Appl.*, vol. 45, no. 3, pp. 543–555, 2010.
[56] D. G. Luenberger, *Linear and Nonlinear Programming*. Reading, MA, USA: Addison-Wesley, 1984.
[57] J. E. Beasley, "OR-library: Distributing test problems by electronic mail," *J. Oper. Res. Soc.*, vol. 41, no. 11, pp. 1069–1072, Nov. 1990.

Mikhail A. Bragin (S'11–M'17) received the B.S. and M.S. degrees in mathematics from Voronezh State University, Voronezh, Russia, in 2004, the M.S. degree in physics and astronomy from the University of Nebraska–Lincoln, Lincoln, NE, USA, in 2006, and the M.S. and Ph.D. degrees in electrical and computer engineering from the University of Connecticut, Storrs, CT, USA, in 2014 and 2016, respectively.

Dr. Bragin is currently an Assistant Research Professor in electrical and computer engineering with the University of Connecticut. His current research interests include mathematical optimization, including power system optimization, grid integration of renewables (wind and solar), energy-based operation optimization of distributed energy systems, and scheduling of manufacturing systems.

Peter B. Luh (S'77–M'80–SM'91–F'95–LF'16) received the B.S. degree from National Taiwan University, Taipei, Taiwan, the M.S. degree from M.I.T., Cambridge, MA, USA, and the Ph.D. degree from Harvard University, Cambridge, MA, USA.

He has been with the University of Connecticut, Storrs, CT, USA, since the 1980s, where he is currently the SNET Professor of Communications and Information Technologies. His current research interests include smart power systems—smart grid, design of auction methods for electricity markets, effective renewable (wind and solar) integration to the grid, electricity load and price forecasting with demand response, and microgrid.

Dr. Luh was the Vice President of Publication Activities for the IEEE Robotics and Automation Society.

Bing Yan (S'11–M'17) received the B.S. degree from the Renmin University of China, Beijing, China, in 2010, and the M.S. and Ph.D. degrees from the University of Connecticut, Storrs, CT, USA, in 2012 and 2016, respectively.

She is currently an Assistant Research Professor with the Department of Electrical and Computer Engineering, University of Connecticut. Her current research interests include power system optimization, grid integration of renewables (wind and solar), energy-based operation optimization of distributed energy systems, and scheduling of manufacturing systems.

Xiaorong Sun (S'12) received the B.S. degree in electrical engineering from Hohai University, Nanjing, China, in 2010, and the M.S. degree from the University of Connecticut, Storrs, CT, USA, in 2014, where she is currently pursuing the Ph.D. degree with the Electrical and Computer Engineering Department.

Her current research interests include electricity load and price forecasting, transmission and distribution systems, and power system optimization.