scientific reports

Check for updates

OPEN

Surrogate "Level-Based" Lagrangian Relaxation for mixed-integer linear programming

Mikhail A. Bragin[1]✉ & Emily L. Tucker[2]

Mixed-Integer Linear Programming (MILP) plays an important role across a range of scientific disciplines and within areas of strategic importance to society. The MILP problems, however, suffer from *combinatorial complexity*. Because of integer decision variables, as the problem size increases, the number of possible solutions increases *super-linearly* thereby leading to a drastic increase in the computational effort. To efficiently solve MILP problems, a "price-based" decomposition and coordination approach is developed to exploit 1. the super-linear reduction of complexity upon the decomposition and 2. the geometric convergence potential inherent to Polyak's stepsizing formula for the fastest coordination possible to obtain near-optimal solutions in a computationally efficient manner. Unlike all previous methods to set stepsizes heuristically by adjusting hyperparameters, the key novel way to obtain stepsizes is purely decision-based: a novel "auxiliary" constraint satisfaction problem is solved, from which the appropriate stepsizes are inferred. Testing results for large-scale Generalized Assignment Problems demonstrate that for the majority of instances, certifiably optimal solutions are obtained. For stochastic job-shop scheduling as well as for pharmaceutical scheduling, computational results demonstrate the two orders of magnitude speedup as compared to Branch-and-Cut. The new method has a major impact on the efficient resolution of complex Mixed-Integer Programming problems arising within a variety of scientific fields.

Mixed-Integer Linear Programming (MILP) plays an important role across a range of scientific disciplines such as mathematics, operations research, engineering, and computer science as well as within a range of areas of strategic importance to society such as biology[1,2], healthcare[3,4], humanitarian applications[5–8], manufacturing[9–12], pharmacy[13–16], power and energy systems[17–19], transportation and logistics[20,21] and many others.

The associated systems are created by interconnecting I smaller subsystems, each having its own objective and a set of constraints. The subsystem interconnection is modeled through the use of *system-wide coupling* constraints. Accordingly, the MILP problems are frequently formulated in terms of cost components associated with each subsystem with the corresponding objective functions being additive as such:

$$\min_{(x,y):=\{x_i,y_i\}_{i=1}^{I}} \left\{ \sum_{i=1}^{I} \left((c_i^x)^T x_i + (c_i^y)^T y_i \right) \right\}. \tag{1}$$

Furthermore, coupling constraints are additive in terms of I subsystems:

$$s.t. \quad \sum_{i=1}^{I} A_i^x x_i + \sum_{i=1}^{I} A_i^y y_i - b = 0, \quad \{x_i, y_i\} \in \mathcal{F}_i, i = 1, \ldots, I. \tag{2}$$

The *primal* problem (1), (2) is assumed to be feasible and the feasible region $\mathcal{F} = \prod_{i=1}^{I} \mathcal{F}_i$ with $\mathcal{F}_i \subset \mathbb{Z}^{n_i^x} \times \mathbb{R}^{n_i^y}$ is assumed to be bounded and finite. The MILP problems modeling the above systems are referred to as *separable*. Because of the discrete decisions, however, MILP problems are known to be NP-hard and are prone to the curse of *combinatorial complexity*. As the size of a problem increases, the associated number of combinations of possible

[1]Department of Electrical and Computer Engineering, University of Connecticut, 371 Fairfield Way, U-4157, Storrs 06269, CT, USA. [2]Department of Industrial Engineering, Clemson University, 271 Freeman Hall, Clemson, SC 29634, USA. ✉email: mikhail.bragin@uconn.edu

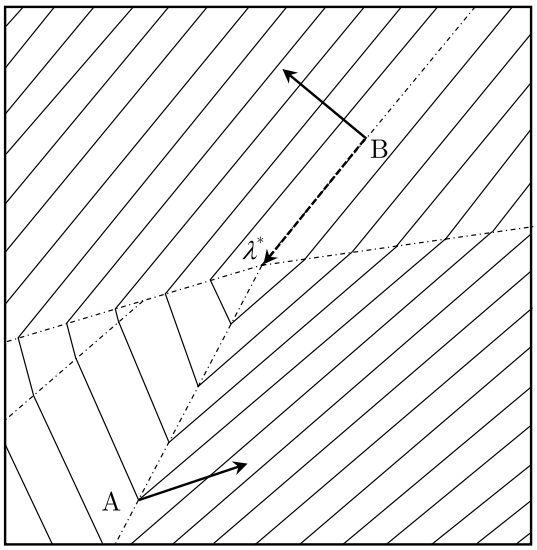

Figure 1. An example of a dual function demonstrating difficulties faced by subgradient methods. Solid lines denote the level curves, dash-dotted lines denote the ridges of the dual function whereby the usual gradients are not defined (possible subgradient directions at points (**A**) and (**B**) are denoted by solid arrows), and the direction from point (**B**) toward optimal multipliers is denoted by a dashed line.

solutions (hence the term "combinatorial") increases super-linearly (e.g., exponentially) thereby making problems of practical sizes difficult to solve to optimality; even near-optimal solutions are frequently difficult to obtain.

A beacon of hope to resolve combinatorial difficulties lies through the exploitation of separability through the *dual* "price-based" decomposition and coordination Lagrangian Relaxation technique. After the relaxation of coupling constraints (2), the coordination of subproblems amounts to the maximization of a concave non-smooth dual function:

$$\max_{\lambda}\{q(\lambda) : \lambda \in \mathbb{R}^m\}, \tag{3}$$

where

$$q(\lambda) = \min_{(x,y)} \left\{ L(x, y, \lambda), \{x_i, y_i\} \in \mathcal{F}_i, i = 1, \ldots, I \right\}. \tag{4}$$

Here $L(x, y, \lambda) \equiv \sum_{i=1}^{I} (c_i^x)^T x_i + \sum_{i=1}^{I} (c_i^y)^T y_i + \lambda^T \cdot \left(\sum_{i=1}^{I} A_i^x x_i + \sum_{i=1}^{I} A_i^y y_i - b \right)$ is the Lagrangian function. The Lagrangian multipliers λ ("dual" variables) are the decision variables with respect to the dual problem (3), and it is assumed that the set of optimal solutions is not empty. The minimization within (4) with respect to $\{x, y\}$ is referred to as the "relaxed problem."

While the sizes of the primal and the relaxed problems are the same in terms of the number of discrete variables, the main advantage of Lagrangian Relaxation is the exploitation of the reduction of the combinatorial complexity upon decomposition into subproblems. Accordingly, the number of discrete decision variables within the primal problem is $n = \sum_{i=1}^{I} n_i^x$, so the worst-case complexity of solving the primal problems is $O(e^{\sum_{i=1}^{I} n_i^x})$. By the same token, the worst-case complexity required to solve the following subproblem

$$\min_{x_i, y_i} \left\{ (c_i^x)^T x_i + (c_i^y)^T y_i + \lambda^T \cdot \left(A_i^x x_i + A_i^y y_i \right), \{x_i, y_i\} \in \mathcal{F}_i \right\}, \tag{5}$$

is $O(e^{n_i^x})$. The decomposition "reverses" the combinatorial complexity thereby exponentially reducing the effort. The decomposition, therefore, offers a viable potential to improve the operations of existing systems as well as to scale up the size of the systems to support their efficient operations.

While decomposition efficiently reduces the combinatorial complexity, the coordination aspect of the method to efficiently obtain the optimal "prices" (Lagrangian multipliers) has been the subject of an intense research debate for decades because of the fundamental difficulties of non-smooth optimization. Namely, because of the presence of integer variables x, the dual function (3) is non-smooth comprised of flat convex polygonal facets (each corresponding to a particular solution to the relaxed problem within (4)) intersecting at linear ridges along which the dual function $q(\lambda)$ is non-differentiable; in particular, $q(\lambda)$ is not differentiable at λ^* thereby ruling out the possibility of using necessary and sufficient conditions for the extremum. As a result of the non-differentiability of $q(\lambda)$, subgradient multiplier-updating directions, however, are non-ascending directions thereby leading to a decrease of dual values; subgradient directions may also change drastically thereby resulting in zigzagging of Lagrangian multipliers (see Fig. 1 for illustrations) and slow convergence as a result.

Traditional methods to maximize $q(\lambda)$ rely upon iterative updates of Lagrangian multipliers by taking a series of steps s^k along subgradient $g(x^k, y^k)$ directions as:

$$\lambda^{k+1} = \lambda^k + s^k \cdot g(x^k, y^k), \tag{6}$$

where $\{x^k, y^k\} \equiv \{x_i^k, y_i^k\}_{i=1}^I$ is a an optimal solution to the relaxed problem (4) with multipliers equal to λ^k. Within the Lagrangian Relaxation framework, subgradients are defined as levels of constraint violations $g(x^k, y^k) \equiv \sum_{i=1}^I A_i^x x_i^k + \sum_{i=1}^I A_i^y y_i^k - b$. Inequality constraints $\sum_{i=1}^I A_i^x x_i + \sum_{i=1}^I A_i^y y_i \le b$, if present, can be handled by converting into equality constraints by introducing non-negative real-valued slack variables z such that $\sum_{i=1}^I A_i^x x_i + \sum_{i=1}^I A_i^y y_i + z = b$. The multipliers are subsequently projected onto the positive orthant delineated by restrictions $\lambda \ge 0$.

Because of the lack of differentiability of $q(\lambda)$, notably, at the optimum λ^*, the stepsize selection plays an important role to guarantee convergence to the optimum as well as for the success of the overall Lagrangian Relaxation methodology for solving MILP problems.

One of the earlier papers on the optimization of non-smooth convex functions, with $q(\lambda)$ being its member, though irrespective of Lagrangian Relaxation, is Polyak's seminal work[22]. Intending to achieve the geometric (also referred to as "linear") rate of convergence so that $\|\lambda^k - \lambda^*\|$ is monotonically decreasing, Polyak proposed the stepsizing formula, which in terms of the problem under consideration takes the following form:

$$0 < s^k < \gamma \cdot \frac{q(\lambda^*) - q(\lambda^k)}{\left\|g(x^k, y^k)\right\|^2}, \gamma < 2. \tag{7}$$

Within (7) and thereafter in the paper the standard Euclidean norm is used.

Subgradient directions, however, 1. are generally difficult to obtain computationally when the number of subproblems (5) to be solved is large, and 2. change drastically thereby resulting in zigzagging of Lagrangian multipliers and slow convergence. Moreover, 3. stepsizes (7) cannot be set due to the lack of the knowledge about the optimal dual value $q(\lambda^*)$.

To overcome the first two of the difficulties above, the Surrogate Subgradient method was developed by[23] whereby the exact optimality of the relaxed problem (or even subproblems) is not required. As long as the following "surrogate optimality condition" is satisfied:

$$L(\tilde{x}^k, \tilde{y}^k, \lambda^k) < L(\tilde{x}^{k-1}, \tilde{y}^{k-1}, \lambda^k) \tag{8}$$

the multipliers can be updated by using the following version of the Polyak's formula

$$0 < s^k < \gamma \cdot \frac{q(\lambda^*) - L(\tilde{x}^k, \tilde{y}^k, \lambda^k)}{\left\|g(\tilde{x}^k, \tilde{y}^k)\right\|^2}, \quad \gamma < 1, \tag{9}$$

and convergence to λ^* is guaranteed. Here "tilde" is used to distinguish optimal solutions $\{x^k, y^k\}$ to the relaxed problem from the solutions $\{\tilde{x}^k, \tilde{y}^k\}$ that satisfy the "surrogate optimality condition" (8). Unlike that in Polyak's formula, parameter γ is less than 1 to guarantee that $q(\lambda^*) > L(\tilde{x}^k, \tilde{y}^k, \lambda^k)$ so that the stepsizing formula (9) is well-defined, as proved by Zhao et al.[23]. Once $\{\tilde{x}^k, \tilde{y}^k\}$ are obtained, multipliers are updated by using the same formula as in (6) with stepsizes from (9) and "surrogate subgradient" multiplier-updating directions $g(\tilde{x}^k, \tilde{y}^k)$ used in place of subgradient directions $g(x^k, y^k)$. Besides reducing the computational effort owing to (8), the concomitant reduction of multiplier zigzagging has also been observed. The main difficulty is the lack of knowledge about $q(\lambda^*)$. As a result, the geometric/linear convergence of the method (or any convergence at all) is highly questionable in practice. Nevertheless, the underlying geometric convergence principle behind the formula (8) is promising and will be exploited in "Results" section.

One of the first attempts to overcome the difficulty associated with the unavailability of the optimal [dual] value is the Subgradient-Level method developed by Goffin and Kiwiel [24] by adaptively adjusting a "level" estimate based on the detection of "sufficient descent" of the [dual] function and "oscillation" of [dual] solutions. In a nutshell, a "level" estimate is set as $q_{lev}^k = q_{rec}^{k_j} + \delta_j$ with q_{rec}^k being the best dual value ("record objective value") obtained up to an iteration k, and δ_j is an adjustable parameter with j denoting the j^{th} update of q_{lev}^k. Every time oscillations of multipliers are detected, δ_j is reduced by half. In doing so, stepsizes appropriately decrease, q_{lev}^k increases (for maximization of non-smooth functions such as (3)) and the process continues until $\delta_j \to 0$ and $q_{lev}^k \to q(\lambda^*)$.

To improve convergence, rather than updating all the multipliers "at once," within the Incremental Subgradient methods[25], multipliers are updated "incrementally." Convergence results of the Subgradient-Level method[24] have been extended for the Incremental Subgradient methods.

Within the Surrogate Lagrangian Relaxation (SLR) method[26], the computational effort is reduced along the lines of the Surrogate Subgradient method[23] discussed above, that is, by solving one of a few subproblems at a time. To guarantee convergence, within SLR, distances between multipliers at consecutive iterations are required to decrease through a specially-constructed contraction mapping until convergence. As demonstrated by Bragin et al.[26], the SLR method converges faster as compared to the above-mentioned Subgradient-Level method[24] and the Incremental Subgradient methods[25,27] for non-smooth optimization. Unlike the Subgradient-Level and Incremental Subgradient methods[25,27], the SLR method does not require obtaining dual values to set stepsizes, which further reduces the effort. Aiming to simultaneously guarantee convergence while ensuring fast

reduction of constraint violations and preserving the linearity, the Surrogate Absolute-Value Lagrangian Relaxation (SAVLR) method[28] was developed to penalize constraint violations by using l_1 "absolute-value" penalty terms. The above methods are reviewed in more detail in Supplementry Information Section.

Because of the presence of the integer variables, there is the so-called the *duality gap*, which means that even at convergence, $q(\lambda^*)$ is generally less than the optimal cost of the original problem (1), (2). To obtain a feasible solution to (1), (2), the subproblem solutions when put together may not satisfy all the relaxed constraints. Therefore, to solve corresponding MILP problems, heuristics are inevitable and are used to perturb subproblem solutions. The important remark here is that the closer the multipliers are to the optimum, generally, the closer the subproblem solutions are to the global optimum of the original problem, and the easier it is to obtain feasible solutions through heuristics. Therefore, having fast convergence in the dual space to maximize the dual function (3) is of paramount importance for the overall success of the method. Specific heuristics will be discussed at the end of the "Results" section.

Results

Surrogate "Level-Based" Lagrangian Relaxation. In this subsection, a novel Surrogate "Level-Based" Lagrangian Relaxation (SLBLR) method is developed to determine "level" estimates of $q(\lambda^*)$ within the Polyak's stepsizing formula (9) for fast convergence of multipliers when optimizing the dual function (3). Since the knowledge of $q(\lambda^*)$ is generally unavailable, over-estimates of the optimal dual value, if used in place of $q(\lambda^*)$ within the formula (9), may lead to the oscillation of multipliers and to the divergence. Rather than using heuristic "oscillation detection" of multipliers used to adjust "level" values[24], the key of SLBLR is the decision-based "divergence detection" of multipliers based on a novel auxiliary "multiplier-divergence-detection" constraint satisfaction problem.

"Multiplier-Divergence-Detection" problem to obtain the estimate of $q(\lambda^*)$. The premise behind the multiplier-divergence detection is the rendition of the result due Zhao et al.[23]:

Theorem 1 *Under the stepsizing formula*

$$s^k < \gamma \cdot \frac{q(\lambda^*) - L(\tilde{x}^k, \tilde{y}^k, \lambda^k)}{\|g(\tilde{x}^k, \tilde{y}^k)\|^2}, \gamma < 1, \tag{10}$$

such that $\{\tilde{x}^k, \tilde{y}^k\}$ *satisfy*

$$L(\tilde{x}^k, \tilde{y}^k, \lambda^k) \leq L(\tilde{x}^{k-1}, \tilde{y}^{k-1}, \lambda^k), \tag{11}$$

the multipliers move closer to optimal multipliers λ^* iteration by iteration:

$$\|\lambda^* - \lambda^{k+1}\| < \|\lambda^* - \lambda^k\|. \tag{12}$$

The following Corollary and Theorem 2 are the main key results of this paper.

Corollary 1 *If*

$$\|\lambda^* - \lambda^{k+1}\| \geq \|\lambda^* - \lambda^k\|, \tag{13}$$

then

$$s^k \geq \gamma \cdot \frac{q(\lambda^*) - L(\tilde{x}^k, \tilde{y}^k, \lambda^k)}{\|g(\tilde{x}^k, \tilde{y}^k)\|^2}. \tag{14}$$

Theorem 2 *If the following auxiliary "multiplier-divergence-detection" feasibility problem (with λ being a continuous decision variable*: $\lambda \in \mathbb{R}^m$)

$$\begin{cases} \|\lambda - \lambda^{k_j+1}\| \leq \|\lambda - \lambda^{k_j}\|, \\ \|\lambda - \lambda^{k_j+2}\| \leq \|\lambda - \lambda^{k_j+1}\|, \\ \ldots \\ \|\lambda - \lambda^{k_j+n_j}\| \leq \|\lambda - \lambda^{k_j+n_j-1}\|, \end{cases} \tag{15}$$

admits no feasible solution with respect to λ for some k_j and n_j, then $\exists\ \kappa \in [k_j, k_j + n_j]$ such that

$$s^\kappa \geq \gamma \cdot \frac{q(\lambda^*) - L(\tilde{x}^\kappa, \tilde{y}^\kappa, \lambda^\kappa)}{\|g(\tilde{x}^\kappa, \tilde{y}^\kappa)\|^2}. \tag{16}$$

Proof Assume the contrary: $\forall \kappa \in [k_j, k_j + n_j]$ the following holds:

$$s^\kappa < \gamma \cdot \frac{q(\lambda^*) - L(\tilde{x}^\kappa, \tilde{y}^\kappa, \lambda^\kappa)}{\|g(\tilde{x}^\kappa, \tilde{y}^\kappa)\|^2}. \tag{17}$$

By Theorem 1, multipliers approach λ^*, therefore, the "multiplier-divergence-detection" problem admits at least one feasible solution - λ^*. Contradiction. □

From (16) it follows that $\exists\ \overline{q}_{\kappa,j}$ such that $\overline{q}_{\kappa,j} > q(\lambda^*)$ and the following holds:

$$s^{\kappa} = \gamma \cdot \frac{\overline{q}_{\kappa,j} - L(\tilde{x}^{\kappa}, \tilde{y}^{\kappa}, \lambda^{\kappa})}{\|g(\tilde{x}^{\kappa}, \tilde{y}^{\kappa})\|^2}. \tag{18}$$

The equation (18) can equivalently be rewritten as:

$$\overline{q}_{\kappa,j} = \frac{1}{\gamma} \cdot s^{\kappa} \cdot \|g(\tilde{x}^{\kappa}, \tilde{y}^{\kappa})\|^2 + L(\tilde{x}^{\kappa}, \tilde{y}^{\kappa}, \lambda^{\kappa}). \tag{19}$$

Therefore,

$$\overline{q}_j = \max_{\kappa \in [k_j, k_j + n_j]} \overline{q}_{\kappa,j} > q(\lambda^*). \tag{20}$$

A brief yet important discussion is in order here. The overestimate $\overline{q}_j$ of the dual value $q(\lambda^*)$ is the sought-for "level" value after the j^{th} update (the j^{th} time the problem (15) is infeasible). Unlike previous methods, which require heuristic hyperparameter adjustments to set level values, within SLBLR, level values are obtained by using the decision-based principle per (15) precisely when divergence is detected without any guesswork. In a sense, SLBLR is hyperparameter-adjustment-free. Specifically, neither "multiplier-divergence-detection" problem (15), nor the computations within (18)–(20) requires hyperparameter adjustment. Following Nedić and Bertsekas[27], the parameter γ will be chosen as a fixed value $\gamma = \frac{1}{I}$, which is the inverse of the number of subproblems and will not require further adjustments.

Note that (15) simplifies to an LP constraint satisfaction problem. For example, after squaring both sides of the first inequality $\|\lambda - \lambda^{k_j+1}\| \leq \|\lambda - \lambda^{k_j}\|$ within (15), after using the binomial expansion, and canceling $\|\lambda - \lambda^{k_j}\|^2$ from both sides, the inequality simplifies to $2 \cdot (\lambda - \lambda^{k_j}) \cdot g(\tilde{x}^{k_j}, \tilde{y}^{k_j}) \geq s^{k_j} \cdot \|g(\tilde{x}^{k_j}, \tilde{y}^{k_j})\|^2$, which is linear in terms of λ.

To speed up convergence, a hyperparameter $\zeta < 1$ is introduced to reduce stepsizes as follows:

$$s^k = \zeta \cdot \gamma \cdot \frac{\overline{q}_j - L(\tilde{x}^k, \tilde{y}^k, \lambda^k)}{\|g(\tilde{x}^k, \tilde{y}^k)\|^2}, \zeta < 1. \tag{21}$$

Subsequently after iteration k_{j+1}, the problem (15) is sequentially solved again by adding one inequality per multiplier-updating iteration until iteration $k_{j+1} + n_{j+1} - 1$ is reached for some n_{j+1} so that (15) is infeasible. Then, stepsize is updated by using $\overline{q}_{j+1}$ per (21) and is used to update multipliers until the next time it is updated to $\overline{q}_{j+2}$ when the "multiplier-divergence-detection" problem is infeasible again, and the process repeats. Per (21), SLBLR requires hyperparameter ζ, yet, it is set before the algorithm is run and subsequently is not adjusted (see "Numerical testing" section for empirical demonstration of the robustness of the method with respect to the choice of hyperparameter ζ).

To summarize the advantage of SLBLR, hyperparameter adjustment is not needed. The guesswork of when to adjust the level-value, and by how much is obviated — after (15) is infeasible, the level value is formulaically recalculated.

On improvement of convergence. To speed up the acceleration of the multiplier-divergence detection through the "multiplier-divergence-detection" problem, (15) is modified, albeit heuristically, in the following way:

$$\begin{cases} \|\lambda - \lambda^{k_j+1}\| \leq \sqrt{1 - 2 \cdot \nu \cdot s^{k_j}} \cdot \|\lambda - \lambda^{k_j}\|, \\ \|\lambda - \lambda^{k_j+2}\| \leq \sqrt{1 - 2 \cdot \nu \cdot s^{k_j+1}} \cdot \|\lambda - \lambda^{k_j+1}\|, \\ \ldots \\ \|\lambda - \lambda^{k_j+n_j}\| \leq \sqrt{1 - 2 \cdot \nu \cdot s^{k_j+n_j-1}} \cdot \|\lambda - \lambda^{k_j+n_j-1}\|. \end{cases} \tag{22}$$

Unlike the problem (15), the problem (22) no longer simplifies to an LP problem. Nevertheless, the system of inequalities delineate the convex region and can still be handled by commercial software.

Discussion of (22). Equation (22) is developed based on the following principles: 1. Rather than detecting divergence per (15), convergence with a rate slower than $\sqrt{1 - 2 \cdot \nu \cdot s}$ is detected. This will lead to a faster adjustment of the level values. While the level value may no longer be guaranteed to be the upper bound to $q(\lambda^*)$, the merit of the above scheme will be empirically justified in the "Numerical testing" section. 2. While the rate of convergence is unknown, in the "worst-case" scenario $\sqrt{1 - 2 \cdot \nu \cdot s}$ is upper bounded by 1 with $\nu = 0$, thereby reducing (22) to (15). The estimation of $\sqrt{1 - 2 \cdot \nu \cdot s}$ is thus much easier than the previously used estimations of $q(\lambda^*)$ (as in Subgradient-Level and Incremental Subgradient approaches). 3. As the stepsize approaches zero, $\sqrt{1 - 2 \cdot \nu \cdot s}$ approaches the value of 1 regardless of the value of ν, once again reducing (22) to (15).

Algorithm: Pseudocode.

Input λ^0, γ, ν, ζ, $\overline{q}_0(> q(\lambda^*))$, $q^{max} = -\infty$

1: **while** $\frac{f(x^{feas},y^{feas})-q(\lambda^k)}{f(x^{feas},y^{feas})} > \varepsilon_{gap}$ **do**
2: solve subproblem (5) s.t. (8),
3: calculate $g(\tilde{x}^k, \tilde{y}^k)$
4: calculate $L(\tilde{x}^k, \tilde{y}^k, \lambda^k)$
5: calculate stepsizes per (21) as $s^k = \zeta \cdot \gamma \cdot \frac{\overline{q}_j - L(\tilde{x}^k,\tilde{y}^k,\lambda^k)}{\|g(\tilde{x}^k,\tilde{y}^k)\|^2}$
6: update multipliers per (6) by using $g(\tilde{x}^k, \tilde{y}^k)$ as $\lambda^{k+1} = \lambda^k + s^k \cdot g(\tilde{x}^k, \tilde{y}^k)$
7: **if** $q^{max} < s^k \frac{\|g(\tilde{x}^k,\tilde{y}^k)\|^2}{\gamma} + L(\tilde{x}^k, \tilde{y}^k, \lambda^k)$ **then** $q^{max} = s^k \frac{\|g(\tilde{x}^k,\tilde{y}^k)\|^2}{\gamma} + L(\tilde{x}^k, \tilde{y}^k, \lambda^k)$
8: **end if**
9: $i \leftarrow i + 1$
10: $k \leftarrow k + 1$
11: **if** $i = I$ **then** $i \leftarrow 1$
12: **end if**
13: **if** (15) is infeasible **then** $\overline{q}_j = q^{max}, q^{max} = -\infty, j \leftarrow j + 1$
14: **end if**
15: **if** $\frac{\overline{q}_j - q(\lambda^k)}{\overline{q}_j} < \varepsilon$ **then** search for feasible solutions x^{feas}, y^{feas} that satisfy (2) to obtain a feasible cost $f(x^{feas}, y^{feas}) \equiv \sum_{i=1}^{I} \left((c_i^x)^T x_i^{feas} + (c_i^y)^T y_i^{feas} \right)$
16: **end if**
17: **end while**

There are three things to note here. 1. Steps in lines 15-16 are optional since other criteria can be used such as the number of iterations or the CPU time; 2. The value of $q(\lambda^k)$ is still needed (line 1) to obtain a valid lower bound. To obtain $q(\lambda^k)$, all subproblems are solved optimally for a given value of multipliers λ^k. The frequency of the search for the value $q(\lambda^k)$ is determined based on criteria as stated in point 1 above; 3. The search for feasible solutions is explained below.

Search for feasible solutions. Due to non-convexities caused by discrete variables, the relaxed constraints are generally not satisfied through coordination, even at convergence. Heuristics are thus inevitable, yet, they are the last step of the feasible-solution search procedure. Throughout all examples considered, following[28] (as discussed in Supplementary Information), l_1-absolute-value penalties penalizing constraint violations are considered. After the total constraint violation reaches a small threshold value, a few subproblem solutions obtained by the Lagrangian Relaxation method are perturbed, e.g., see heuristics within accompanying CPLEX codes within[28] to automatically select which subproblem solutions are to be adjusted to eliminate the constraint violation to obtain a solution feasible with respect to the overall problem.

Numerical Testing. In this subsection, a series of examples are considered to illustrate different aspects of the SLBLR method. In "Demonstration of convergence of multipliers based on a small example with known optimal multipliers" section, a small example with known corresponding optimal Lagrangian multipliers is considered to test the new method as well as to compare how fast Lagrangian multipliers approach their optimal values as compared to Surrogate Lagrangian Relaxation[26] and to Incremental Subgradient[25] methods. In "Generalized Assignment Problems" section, large-scale instances of generalized assignment problems (GAPs) of types D and E with 20, 40, and 80 machines and 1600 jobs from the OR-library (https://www-or.amp.i.kyoto-u.ac.jp/members/yagiura/gap/) are considered to demonstrate efficiency, scalability, robustness, and competitiveness of the method with respect to the best results available thus far in the literature. In "Stochastic job-shop scheduling with the considerationof scrap and rework" section, a stochastic version of a job-shop scheduling problem instance with 127 jobs and 19 machines based on Hoitomt et al.[29] is tested. In "Multi-stage pharmaceutical scheduling" section, two instances of pharmaceutical scheduling with 30 and 60 product orders, 17 processing units, and 6 stages based on Kopanos et al.[13] are tested.

For "Demonstration of convergence of multipliers based on a small example with known optimal multipliers" section and "Generalized Assignment Problems" section, SLBLR is implemented within CPLEX 12.10 by using a Dell Precision laptop Intel(R) Xeon(R) E-2286M CPU @ 2.40GHz with 16 cores and installed memory (RAM) of 32.0 GB. For "Stochastic job-shop scheduling with the considerationof scrap and rework" section and "Multi-stage pharmaceutical scheduling" section, SLBLR is implemented within CPLEX 12.10 by using a server Intel(R) Xeon(R) Gold 6248R CPU @ 3.00GHz with 48 cores and installed memory (RAM) of 192.0 GB.

Demonstration of convergence of multipliers based on a small example with known optimal multipliers. To demonstrate convergence of multipliers, consider the following example (due Bragin et al.[30]):

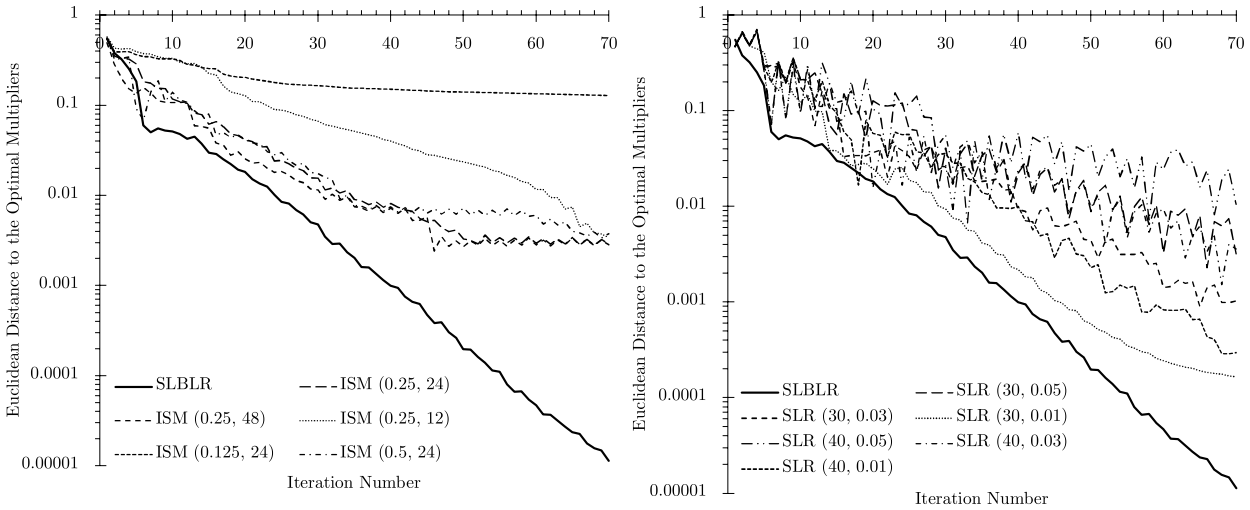

Figure 2. Results for "Demonstration of convergence of multipliers based on a small example with known optimal multipliers" section: Comparison of SLBLR to 1. Incremental Subgradient method (left) and 2. Surrogate Lagrangian Relaxation method (right).

$$\min_{x_1,x_2,x_3,x_4,x_5,x_6} \{x_1 + 2x_2 + 3x_3 + x_4 + 2x_5 + 3x_6\}, \tag{23}$$

$$s.t.\ x_1 + 3x_2 + 5x_3 + x_4 + 3x_5 + 5x_6 \geq 26, \tag{24}$$

$$2x_1 + 1.5x_2 + 5x_3 + 2x_4 + 0.5x_5 + x_6 \geq 16. \tag{25}$$

As proved by Bragin et al.[30], the optimal dual solutions are $\lambda_1^* = 0.6$ and $\lambda_2^* = 0$. Inequality constraints are converted to equality constraints after introducing slack variables. In Fig. 2, the decrease of the corresponding distances from current multipliers to the optimal multipliers ($\|\lambda^k - \lambda^*\|$) is shown, and the SLBLR method is compared with the Incremental Subgradient method[25] and the Surrogate Lagrangian Relaxation method[26].

Within the SLBLR method, the equation (15) is used to detect divergence, and $\zeta = \frac{1}{2}$ is used to set stepsizes within (21). In essence, only one hyperparameter was required, which has a quite simple explanation - "when the stepsize is 'too large,' cut the stepsize in half." As demonstrated in Fig. 2, the SLBLR method converges fast with $\|\lambda^k - \lambda^*\|$ decreasing roughly along a straight line on a log-scale graph suggesting that the rate of convergence is likely linear as expected.

As for the Incremental Subgradient method, two hyperparameters are required: R and δ (corresponding values used are shown in parentheses in the legend of Fig. 2 (left)). A trial-and-error analysis indicated that "acceptable" values are $R = 0.25$ and $\delta = 24$. Increasing or decreasing R to 0.5 and 0.125, respectively, do not lead to improvements. Likewise, increasing or decreasing δ to 48 and 12, respectively, do not lead to improvements as well. "Plateau" regions in the figure are caused by the fact that as stepsizes get smaller, a larger number of iterations is required for multipliers to travel the predetermined distance R; during these iterations, stepsizes are not updated and multipliers may oscillate around a neighborhood of the optimum without getting closer. While the above difficulty can be alleviated and convergence can be improved by hyperparameters τ, β, and R_l as reviewed in Supplementary Information, however, an even larger number of hyperparameters would be required.

As for the Surrogate Lagrangian Relaxation method, several pairs of hyperparameters (M and r) have been used as well (corresponding values used are shown in parentheses in the legend of Fig. 2 (right)), yet, the performance of Surrogate Lagrangian Relaxaton does not exceed the performance of the SLBLR method.

Herein lies the advantage of the novel SLBLR method: the decision-based principle behind computing the "level" values. This is in contrast to the problem-dependent choice of hyperparameters R and δ within the Subgradient-Level[24] and Incremental Subgradient[25] methods, and the choice of M and r within Surrogate Lagrangian Relaxation[26,28] (see "Introduction" section and Supplementry Information for more detail).

Even after obtaining "appropriate" values of the aforementioned hyperparameters by using a trial-and-error procedure that entails effort, results obtained by Surrogate Lagrangian Relaxation[26] and the Incremental Subgradient method[25] do not match or beat those obtained by the SLBLR method. The specific reasons are 1. Heuristic adjustments of the "level" values are required[24,25] based on multiplier "oscillation detection" or "significant descent" (for minimization of non-smooth functions). However, these rules do not detect whether multipliers "start diverging." Moreover, oscillation of multipliers is a natural phenomenon when optimizing non-smooth functions as discussed in "Introduction" section since multipliers may zigzag/oscillate across ridges of the function, so the multiplier "oscillation detection" may not necessarily warrant the adjustment of level values. On the other hand, multiplier "oscillation" is detected by checking whether multipliers traveled a (heuristically) predetermined distance R, hence, the divergence of multipliers can go undetected for a significant number of iterations (hence, the "plateau" regions shown in Fig. 2 (left)), depending on the value of R. To the best of the

ζ	Feasible cost	Gap (%)	"Auxiliary" time (sec)	Total time (sec)
1/1.25	97827	0.0059	4.59	2904.02
1/1.5	**97825**	0.0037	17.10	1195.36
1/2	**97825**	0.0048	88.59	2612.48
1/4	97827	0.0059	89.01	10235.50

Table 1. Robustness results for instance d201600 with respect to ζ. The best feasible cost values obtained are in bold.

ν	Feasible cost	Gap (%)	"Auxiliary" time (sec)	Total time (sec)
0.03125	97826	0.0048	93.79	2716.68
0.125	**97825**	0.0037	33.62	1820.96
0.5	97826	0.0048	9.61	2444.46
2	**97825**	0.0037	17.10	1195.36

Table 2. Robustness results for instance d201600 with respect to ν. The best feasible cost values obtained are in bold.

Initial stepsize (s^0)	Feasible cost	Gap (%)	"Auxiliary" time (sec)	Total time (sec)
0.0025	**97825**	0.0037	123.71	2427.71
0.005	**97825**	0.0037	6.84	1226.17
0.01	97826	0.0048	6.96	2143.58
0.02	**97825**	0.0037	17.10	1195.36
0.04	97826	0.0048	19.21	1941.55

Table 3. Robustness results for instance d201600 with respect to initial stepsizes s^0. The best feasible cost values obtained are in bold.

authors' knowledge, the subgradient- and surrogate-subgradient-based methods using Polyak's stepsizes with the intention of achieving the geometric/linear convergence rate either require $q(\lambda^*)$, which is unavailable, or require multipliers to travel infinite distance to guarantee convergence to the optimum λ^*[24]. 2. While SLR avoids the need to estimate $q(\lambda^*)$, the geometric/linear convergence is only possible outside of a neighborhood of λ^*[26]. Precisely for this reason, the convergence of multipliers within SLR with the corresponding stepsizing parameters $M = 30$ and $r = 0.01$ (as shown in Fig. 2 (right)) appears to follow closely convergence within SLBLR up until iteration 50, after which the improvement tapers off.

Generalized assignment problems. To demonstrate the computational capability of the new method as well as to determine appropriate values for key hyperparameters ζ and ν while using standard benchmark instances, large-scale instances of GAPs are considered (formulation is available in subsection 4.2 of Supplementary Information). We consider 20, 40, and 80 machines with 1600 jobs (https://www-or.amp.i.kyoto-u.ac.jp/members/yagiura/gap/).

To determine values for ζ within (21) and ν within (22) to be used throughout the examples, several values are tested using GAP instance d201600. In Table 1, with fixed values of $\nu = 2$ and $s^0 = 0.02$, the best result (both in terms of the cost and the CPU time) is obtained with $\zeta = 1/1.5$. With the value of $\zeta = 1/4$, the stepsize decreases "too fast" thereby leading to a larger number of iterations and a much-increased CPU time as a result. Likewise, in Table 2 with fixed values of $\zeta = 1/1.5$ and $s^0 = 0.02$, it is demonstrated that the best result (both in terms of the cost and the CPU time) is obtained with $\nu = 2$. Empirical evidence here suggests that the method is stable for other values of ν. The robustness with respect to initial stepsizes (s^0) is tested and the results are demonstrated in Table 3. Multipliers are initialized by using LP dual solutions. The method's performance is appreciably stable for the given range of initial stepsizes used (Table 3). SLBLR is robust with respect to initial multipliers λ^0 (Table 4). For this purpose, the multipliers are initialized randomly by using the uniform distribution $U[90, 110]$. For the testing, the initial stepsize $s^0 = 0.02$ was used. As evidenced from Table 4, the method's performance is stable, exhibiting only a slight degradation of solution accuracy and an increase of the CPU time as compared to the case with multipliers initialized by using LP dual solutions.

To test the robustness as well as scalability of the method across several large-scale GAP instances, six instances d201600, d401600, d801600, e201600, e401600, and e801600 are considered. SLBLR is compared with Depth-First Lagrangian Branch-and-Bound method (DFLBnB)[31], Column Generation[32], and Very Large Scale Neighborhood Search (VLSNS)[33], which to the best of the authors' knowledge are the best methods for at least one of the above instances. For completeness, a comparison against Surrogate Absolute-Value Lagrangian Relaxation (SAVLR)[28], which is an improved version of Surrogate Lagrangian Relaxation (SLR)[26], is also performed. The latter SLR method[26] has been previously demonstrated to be advantageous against other non-smooth optimization

Case number	Feasible cost	Total subproblem solving time (sec)	Feasible solution search time (sec)	"Auxiliary" time (sec)	Total time (sec)
1	**97825**	1098.74	375.96	22.13	1496.84
2	97826	1009.42	777.16	173.48	1960.07
3	97826	2223.99	221.70	4.54	2450.24
4	97826	2333.55	402.41	4.08	2740.04
5	97826	1002.77	119.91	160.73	1283.42

Table 4. Robustness results for instance d201600 with respect to initial multipliers λ^0. The best feasible cost values obtained are in bold.

Instance	New method (SLBLR)	Posta[31] (DFLBnB)	Sadykov[32] (Column generation)	Haddadi[33] (VLSNS)	Bragin[28] (SAVLR)
d201600	**97825** (1195)	– †	**97825** (1026)	97836 (5364)	97828 (1371)
d401600	**97105*** (836)	– †	97106 (919)	97125 (5364)	97111 (1183)
d801600	**97034*** (3670)	– †	97037 (10860)	97075 (5364)	97039 (1350)
e201600	**180645**** (85)	**180645** (40)	–	**180645** (749)	–
e401600	**178293**** (2478)	**178293** (243)	–	**178293** (749)	–
e801600	**176820**** (1762)	**176820** (75)	–	176821 (749)	–

Table 5. Comparison against the best results currently available. * The optimality is certified by the LP optimal values, which are 97105 and 97034 for instances d401600 and d801600, respectively. ** The optimality is certified through the lower bound results of, i.e., Posta et al.[31]. –† Not solved to optimality within 24 hours and not reported within the original paper by Posta et al.[31]. – These instances were not considered within the papers by Sadykov et al.[32] and Bragin et al.[28]. The best feasible cost values obtained are in bold.

methods as explained in "Introduction" section. Table 5 presents feasible costs and times (in seconds) for each method. The advantage of SLBLR is the ability to obtain optimal results across a wider range of GAP instances as compared to other methods. Even though the comparison in terms of the CPU time is not entirely fair, feasible-cost-wise, SLBLR decisively beats previous methods. For the d201600 instance, the results obtained by SLBLR and the Column Generation method[32] are comparable. For instance d401600, SLBLR obtains a better feasible solution and for instance d801600, the advantage over the existing methods is even more pronounced.

To the best of the authors' knowledge, no other reported method obtained optimal results for instances d401600 and d801600. SLBLR outperforms SAVLR[28] as well, thereby demonstrating that the fast convergence offered by the novel "level-based" stepsizing, with other things being equal, translates into better results as compared to those obtained by SAVLR, which employs the "contraction mapping" stepsizing[28]. Lastly, the methods developed in[31–33] specifically target GAPs, whereas the SLBLR method developed in this paper has broader applicability.

Stochastic job-shop scheduling with the consideration of scrap and rework. To demonstrate the computational capability of the method to solve large-scale stochastic MILP problems, a job-shop scheduling problem is considered. Within a job shop, each job requires a specific sequence of operations and the processing time for each operation. Operations are performed by a set of eligible machines. To avoid late shipments, expected tardiness is minimized. Limited machine capacity brings a layer of difficulty since multiple "individual-job" subproblems are considered together competing for limited resources (machines). Another difficulty arises because of uncertainties, including processing times[34–39] and scrap[40–42]. Re-manufacturing of one part may affect and disrupt the overall schedule within the entire job shop, thereby leading to unexpectedly high delays in production.

In this paper, we modified data from the paper by Hoitomt et al.[29] by modifying several jobs by increasing the number of operations (e.g., from 1 to 6) and decreasing the capacities of a few machines; the data are in Tables S1 and S2. The stochastic version of the problem with the consideration of scrap and rework is available within the manuscript by Bragin et al.[42]. With these changes, the running time of CPLEX spans multiple days as demonstrated in Fig. 3. In contrast, within the new method, a solution of the same quality as that obtained by CPLEX, is obtained within roughly 1 hour of CPU time. The new method is operationalized by relaxing machine capacity constraints[42] and coordinating resulting job subproblems; at convergence, the beginning times of several jobs are adjusted by a few time periods to remove remaining machine capacity constraint violations.

Multi-stage pharmaceutical scheduling. To demonstrate the capability of the method to solve scheduling problems complicated by the presence of sequence-dependent setup times, a multi-stage pharmaceutical scheduling problem proposed by Kopanos et al.[13] is considered. Setup times vary based on the sequencing of products on each unit (machine). Scheduling in this context is combinatorial in the number of product orders, units, and stages. The new method is operationalized by relaxing constraints that couple individual processing units,

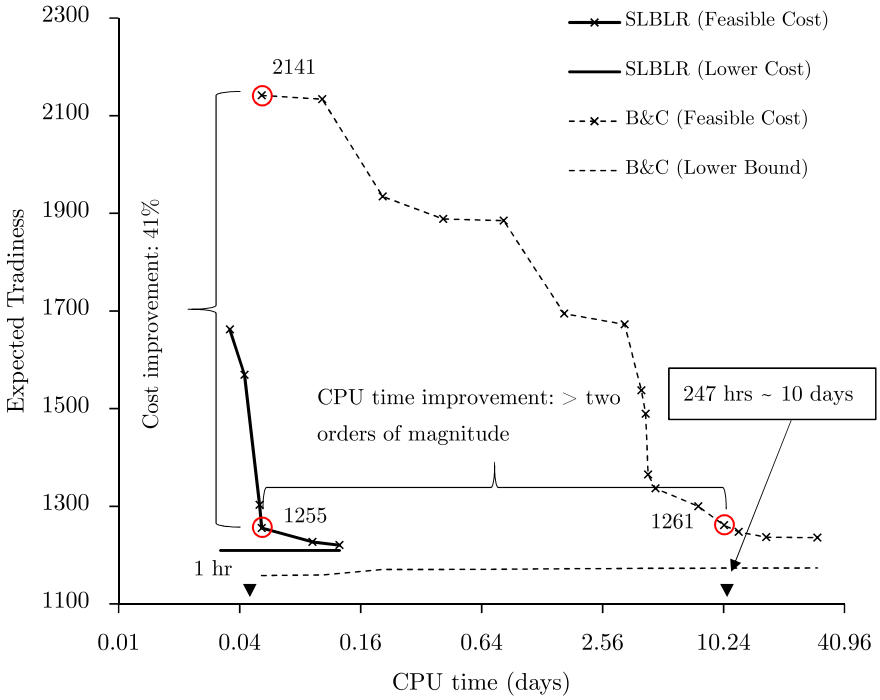

Figure 3. The results for "Stochastic job-shop scheduling with the considerationof scrap and rework" section are illustrated. SLBLR performs more than two orders of magnitude faster than branch-and-cut implemented in CPLEX.

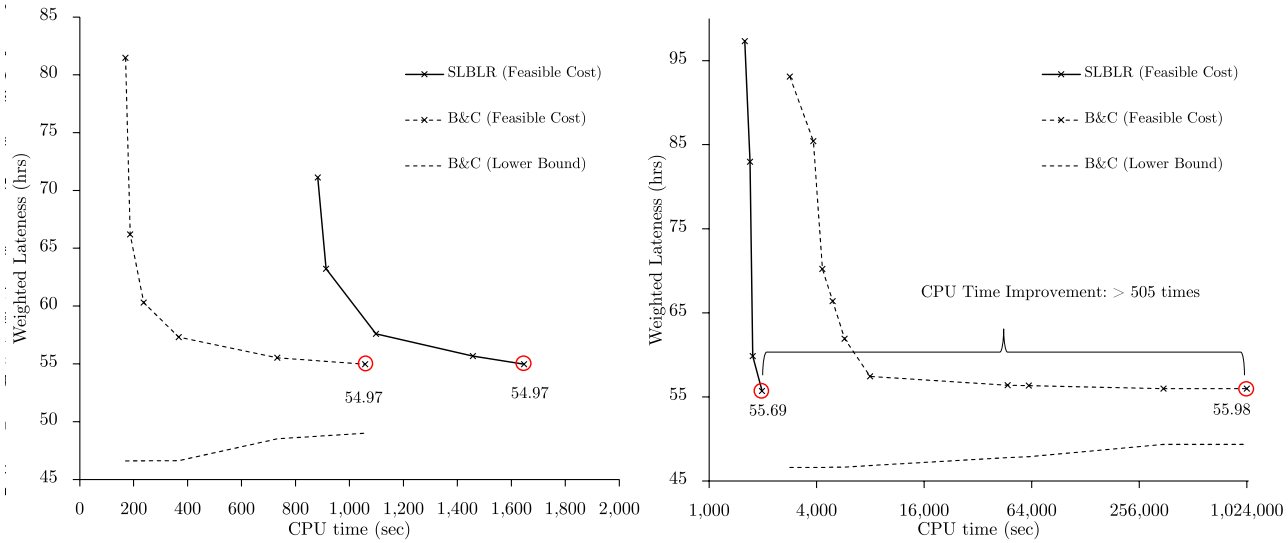

Figure 4. The results for "Multi-stage pharmaceutical scheduling" section with 30 products orders (left) and 60 product orders (right) are illustrated.

namely assignment, and processing/setup time constraints (constraints (39)-(41) from Supplementary Information). The results obtained by SLBLR and Branch-and-Cut are demonstrated in Fig. 4.

With a relatively small number of product orders, 30, an optimal solution with a feasible cost of 54.97 was found by CPLEX within 1057.78 seconds. The optimality is verified by running CPLEX until the gap is 0%; it took 171993.27 seconds to verify the optimality. SLBLR takes a slightly longer time to obtain the same solution - 1647.35 seconds (Fig. 4 (left)). In contrast, with 60 product orders, CPLEX no longer obtains good solutions in a computationally efficient manner; a solution with a feasible cost of 55.98 is obtained after 1,000,000 seconds. Within SLBLR, a solution with a feasible cost of 55.69 is obtained within 1978.04 seconds. This constitutes more than two orders of magnitude of improvement over CPLEX as demonstrated in Fig. 4 (right; log scale). When

doubling the number of products, CPLEX's performance is drastically deteriorated, while the performance of SLBLR is scalable.

Discussion

This paper develops a novel MILP solution methodology based on the Lagrangian Relaxation method. Salient features of the novel SLBLR method, inherited from the previous versions of Lagrangian Relaxation, are: 1. reduction of the computational effort required to obtain Lagrangian-multiplier-updating directions and 2. alleviation of zigzagging of multipliers. The key novel feature of the method, which the authors believe gives SLBLR the decisive advantage, is the innovative exploitation of the underlying geometric-convergence potential inherent to Polyak's stepsizing formula without the heuristic adjustment of hyperparameters for the estimate of $q(\lambda^*)$ - the associated "level" values are determined purely through the simple auxiliary "multiplier-divergence-detection" constraint satisfaction problem. Through testing, it is discovered that SLBLR is robust with respect to the choice of initial stepsizes and multipliers, computationally efficient, competitive, and general. Several problems from diverse disciplines are tested and the superiority of SLBLR is demonstrated. While "separable" MILP problems are considered, no particular problem characteristics such as linearity or separability have been used to obtain "level" values, and thus SLBLR has the potential to solve a broad class of MIP problems.

Data availability

Data supporting the results of "Generalized Assignment Problems" section are located at https://www-or.amp.i.kyoto-u.ac.jp/members/yagiura/gap/; for "Stochastic job-shop scheduling with the considerationof scrap and rework" section, data are located in Tables S1 and S2 as well as in Supplementry Information; for "Multi-stage pharmaceutical scheduling" section, data are taken from the paper by Kopanos et al.[13].

Received: 6 September 2022; Accepted: 13 December 2022
Published online: 27 December 2022

References

1. Huang, P. S., Boyken, S. E. & Baker, D. The coming of age of de novo protein design. *Nature* **537**, 320–327. https://doi.org/10.1038/nature19946 (2016).
2. Yang, L., Chen, R., Goodison, S. & Sun, Y. An efficient and effective method to identify significantly perturbed subnetworks in cancer. *Nat. Comput. Sci.* **1**, 79–88. https://doi.org/10.1038/s43588-020-00009-4 (2021).
3. Khlif Hachicha, H. & Zeghal Mansour, F. Two-MILP models for scheduling elective surgeries within a private healthcare facility. *Health Care Manag. Sci.* **21**(3), 376–392. https://doi.org/10.1007/s10729-016-9390-2 (2018).
4. Kayvanfar, V., Akbari Jokar, M. R., Rafiee, M., Sheikh, S. & Iranzad, R. A new model for operating room scheduling with elective patient strategy. *INFOR Inf. Syst. Oper. Res.* **59**(2), 309–332. https://doi.org/10.1080/03155986.2021.1881359 (2021).
5. Smalley, H. K., Keskinocak, P., Swann, J. & Hinman, A. Optimized oral cholera vaccine distribution strategies to minimize disease incidence: A mixed integer programming model and analysis of a Bangladesh scenario. *Vaccine* **33**(46), 6218–6223. https://doi.org/10.1016/j.vaccine.2015.09.088 (2015).
6. Hamdan, B. & Diabat, A. Robust design of blood supply chains under risk of disruptions using Lagrangian relaxation. *Transp. Res. Part E Logist. Transp. Rev.* **134**, 101764. https://doi.org/10.1016/j.tre.2019.08.005 (2020).
7. Ahani, N., Andersson, T., Martinello, A., Teytelboym, A. & Trapp, A. C. Placement optimization in refugee resettlement. *Oper. Res.* **69**(5), 1468–1486. https://doi.org/10.1287/opre.2020.2093 (2021).
8. Kamyabniya, A., Noormohammadzadeh, Z., Sauré, A. & Patrick, J. A robust integrated logistics model for age-based multi-group platelets in disaster relief operations. *Transp. Res. Part E Logist. Transp. Rev.* **152**, 102371 (2021).
9. Liu, Q., Li, X. & Gao, L. Mathematical modeling and a hybrid evolutionary algorithm for process planning. *J. Intell. Manuf.* **32**(2), 781–797. https://doi.org/10.1007/s10845-020-01703-w (2021).
10. Hong, I.-H., Chou, C.-C. & Lee, P.-K. Admission control in queue-time loop production-mixed integer programming with Lagrangian relaxation (MIPLAR). *Comput. Ind. Eng.* **129**, 417–425. https://doi.org/10.1016/j.cie.2019.02.002 (2019).
11. Balogh, A., Garraffa, M., O'Sullivan, B. & Salassa, F. MILP-based local search procedures for minimizing total tardiness in the No-idle Permutation Flowshop problem. *Comput. Oper. Res.*https://doi.org/10.1016/j.cor.2022.105862 *(2022).*
12. Öztop, H., Tasgetiren, M. F., Kandiller, L. & Pan, Q. K. Metaheuristics with restart and learning mechanisms for the no-idle flowshop scheduling problem with makespan criterion. *Comput. Oper. Res.* **138**, 105616. https://doi.org/10.1016/j.cor.2021.105616 (2022).
13. Kopanos, G. M., Méndez, C. A. & Puigjaner, L. MIP-based decomposition strategies for large-scale scheduling problems in multiproduct multistage batch plants: A benchmark scheduling problem of the pharmaceutical industry. *Eur. J. Oper. Res.* **207**(2), 644–655. https://doi.org/10.1016/j.ejor.2010.06.002 (2010).
14. Stefansson, H., Sigmarsdottir, S., Jensson, P. & Shah, N. Discrete and continuous time representations and mathematical models for large production scheduling problems: A case study from the pharmaceutical industry. *Eur. J. Oper. Res.* **215**(2), 383–392. https://doi.org/10.1016/j.ejor.2011.06.021 (2011).
15. Zhu, S. X. & Ursavas, E. Design and analysis of a satellite network with direct delivery in the pharmaceutical industry. *Transp. Res. Part E Logist. Transp. Rev.* **118**, 190–207. https://doi.org/10.1016/j.tre.2018.06.005 (2018).
16. Ge, C. & Yuan, Z. Production scheduling for the reconfigurable modular pharmaceutical manufacturing processes. *Comput. Chem. Eng.* **151**, 107346. https://doi.org/10.1016/j.compchemeng.2021.107346 (2021).
17. Schill, W.-P., Pahle, M. & Gambardella, C. Start-up costs of thermal power plants in markets with increasing shares of variable renewable generation. *Nat. Energy* **2**(6), 1–6. https://doi.org/10.1038/nenergy.2017.50 (2017).
18. Chen, Y. *et al.* A high performance computing based market economics driven neighborhood search and polishing algorithm for security constrained unit commitment. *IEEE Trans. Power Syst.* **36**(1), 292–302. https://doi.org/10.1109/TPWRS.2020.3005407 (2020).
19. Li, X., Zhai, Q. & Guan, X. Robust transmission constrained unit commitment: A column merging method. *IET Gener. Transm. Distrib.* **14**(15), 2968–2975. https://doi.org/10.1049/iet-gtd.2018.6314 (2020).
20. Archetti, C., Peirano, L. & Speranza, M. G. Optimization in multimodal freight transportation problems: A survey. *Eur. J. Oper. Res.* **299**(1), 1–20. https://doi.org/10.1016/j.ejor.2021.07.031 (2022).
21. Reddy, K. N., Kumar, A., Choudhary, A. & Cheng, T. C. E. Multi-period green reverse logistics network design: An improved Benders-decomposition-based heuristic approach. *Eur. J. Oper. Res.*https://doi.org/10.1016/j.ejor.2022.03.014 *(2022).*
22. Polyak, B. T. Minimization of unsmooth functionals. *USSR Comput. Math. Math. Phys.* **9**(3), 14–29. https://doi.org/10.1016/0041-5553(69)90061-5 (1969).

23. Zhao, X., Luh, P. B. & Wang, J. Surrogate gradient algorithm for Lagrangian relaxation. *J. Optim. Theory Appl.* **100**(3), 699–712. https://doi.org/10.1023/A:1022646725208 (1999).
24. Goffin, J.-L. & Kiwiel, K. C. Convergence of a simple subgradient level method. *Math. Program.* **85**, 207–211. https://doi.org/10.1007/s101070050053 (1999).
25. Nedić, A. & Bertsekas, D. P. Incremental subgradient methods for nondifferentiable optimization. *SIAM J. Optim.* **12**(1), 109–138. https://doi.org/10.1137/S1052623499362111 (2001).
26. Bragin, M. A., Luh, P. B., Yan, J. H., Yu, N. & Stern, G. A. Convergence of the surrogate Lagrangian relaxation method. *J. Optim. Theory Appl.* **164**(1), 173–201. https://doi.org/10.1007/s10957-014-0561-3 (2015).
27. Nedić, A. & Bertsekas, D. Convergence rate of incremental subgradient Algorithms. In *Stochastic Optimization: Algorithms and Applications* Vol. 54 (eds Uryasev, S. & Pardalos, P. M.) (Springer, Boston, 2001).
28. Bragin, M. A., Luh, P. B., Yan, B. & Sun, X. A scalable solution methodology for mixed-integer linear programming problems arising in automation. *IEEE Trans. Autom. Sci. Eng.* **16**(2), 531–541. https://doi.org/10.1109/TASE.2018.2835298 (2019).
29. Hoitomt, D. J., Luh, P. B. & Pattipati, K. R. A practical approach to job shop scheduling problems. *IEEE Trans. Robot. Autom.* **9**(1), 1–13. https://doi.org/10.1109/70.210791 (1993).
30. Bragin, M. A., Yan, B. & Luh, P. B. Distributed and asynchronous coordination of a mixed-integer linear system via surrogate Lagrangian relaxation. *IEEE Trans. Autom. Sci. Eng.* **18**(3), 1191–1205. https://doi.org/10.1109/TASE.2020.2998048 (2020).
31. Posta, M., Ferland, J. A. & Philippe, M. An exact method with variable fixing for solving the generalized assignment problem. *Comput. Optim. Appl.* **52**(3), 629–644. https://doi.org/10.1007/s10589-011-9432-0 (2012).
32. Sadykov, R., Vanderbeck, F., Pessoa, A., & Uchoa, E. Column generation based heuristic for the generalized assignment problem. XLVII Simpósio Brasileiro de Pesquisa Operacional, Porto de Galinhas, Brazil, 3624–3631 (2015).
33. Haddadi, S. Variable-fixing then subgradient optimization guided very large scale neighborhood search for the generalized assignment problem. *4OR Q. J. Oper. Res.* **17**(3), 261–295. https://doi.org/10.1007/s10288-018-0389-z (2019).
34. Golenko-Ginzburg, D. & Gonik, A. Optimal job-shop scheduling with random operations and cost objectives. *Int. J. Prod. Econ.* **76**(2), 147–157. https://doi.org/10.1016/S0925-5273(01)00140-2 (2002).
35. Lei, D. Simplified multi-objective genetic algorithms for stochastic job shop scheduling. *Appl. Soft Comput.* **11**(8), 4991–4996. https://doi.org/10.1016/j.asoc.2011.06.001 (2011).
36. Zhang, R., Song, S. & Wu, C. A hybrid differential evolution algorithm for job shop scheduling problems with expected total tardiness criterion. *Appl. Soft Comput.* **13**(3), 1448–1458. https://doi.org/10.1016/j.asoc.2012.02.024 (2013).
37. Shen, J. & Zhu, Y. Chance-constrained model for uncertain job shop scheduling problem. *Soft Comput.* **20**(6), 2383–2391. https://doi.org/10.1007/s00500-015-1647-z (2016).
38. Jamili, A. Job shop scheduling with consideration of floating breaking times under uncertainty. *Eng. Appl. Artif. Intell.* **78**, 28–36. https://doi.org/10.1016/j.engappai.2018.10.007 (2019).
39. Horng, S. C. & Lin, S. S. Apply ordinal optimization to optimize the job-shop scheduling under uncertain processing times. *Arab. J. Sci. Eng.* **47**, 9659–9671. https://doi.org/10.1007/s13369-021-06317-9 (2022).
40. Wilson, J. P., Shen, Z., Awasthi, U., Bollas, G. M. & Gupta, S. Multi-objective optimization for cost-efficient and resilient machining under tool wear. *J. Adv. Manuf. Process.*https://doi.org/10.1002/amp2.10140 *(2022).*
41. Sun, Y., Tu, J., Bragin, M. A. & Zhang, L. A simulation-based integrated virtual testbed for dynamic optimization in smart manufacturing systems. *J. Adv. Manuf. Process.*https://doi.org/10.1002/amp2.10141 *(2022).*
42. Bragin, M. A., Wilhelm, M. E., & Stuber, M. D. Toward agile and robust supply chains: A lesson from stochastic job-shop scheduling. Preprint at arxiv:2206.09326v1 (2022).

Acknowledgements

The work by M.A.B. was supported in part by the US NSF under award ECCS-1810108.

Author contributions

M.A.B. conceptualized the project, developed the novel methodology, and conducted experiments. E.L.T. developed the fourth case study and supported testing and interpretation of results. M.A.B. drafted the initial manuscript with revisions from E.L.T. Both authors provided feedback on the final version. All authors read and approved the final manuscript.

Competing interests

The authors declare no competing interests.

Additional information

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1038/s41598-022-26264-1.

Correspondence and requests for materials should be addressed to M.A.B.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.