Enhancing large neighbourhood search heuristics for Benders' decomposition

Stephen J. Maher

Department of Management Science, Lancaster University,
Bailrigg, Lancaster LA1 4YX, UK

Abstract

A general enhancement of the Benders' decomposition (BD) algorithm can be achieved through the improved use of large neighbourhood search heuristics within mixed-integer programming solvers. While mixed-integer programming solvers are endowed with an array of large neighbourhood search heuristics, few, if any, have been designed for BD. Further, typically the use of large neighbourhood search heuristics is limited to finding solutions to the BD master problem. Given the lack of general frameworks for BD, only ad hoc approaches have been developed to enhance BD through the use of large neighbourhood search heuristics. The general BD framework of SCIP has been extended with a trust region based heuristic and a general enhancement for all large neighbourhood search heuristics. The general enhancement employs BD to solve the auxiliary problems of all large neighbourhood search heuristics to improve the quality of the identified solutions. The computational results demonstrate the improvements to the heuristic performance of BD achieved by the trust region heuristic and a general large neighbourhood search enhancement technique.

Key words: Benders' decomposition, large neighbourhood search, enhancement techniques, mixed integer programming

1 Introduction

Benders' decomposition (BD) [4] is a popular mathematical programming technique that is used to solve large-scale optimisation problems. Such problems typically arise from industrial contexts, for example airline planning [11, 25, 27], maintenance scheduling [9], production planning [26] or network design [12], or in the form of stochastic programming problems [7, 8, 26, 31]. Large-scale optimisation problems are typically difficult to solve and it may not be always possible to find an optimal solution. As such, BD can be applied as a heuristic approach to find high quality, implementable solutions with provable bounds.

The ability of the BD algorithm to find high quality solutions relies on the algorithmic framework and the available enhancement techniques. Additionally, the underlying mixed integer programming (MIP) solver and the included heuristics are critical to the performance of the BD algorithm. This is particularly important when implementing BD using the branch-and-cut approach,

i.e. using callback functions available within a MIP solver, where more interaction with the solver is supported. While the branch-and-cut approach to BD provides more flexibility in the algorithm implementation, design decisions can significantly affect the algorithmic performance. The appropriate enhancement techniques must be selected to achieve the best heuristic performance from the BD algorithm.

Large neighbourhood search (LNS) heuristics [2] are a valuable tool for finding good quality solutions for difficult MIPs. In addition, previous work has demonstrated their potential for enhancing the BD algorithm [8,28]. While MIP solvers are endowed with many LNS heuristics, their effectiveness when used within decomposition algorithms—such as Dantzig-Wolfe reformulation [14] or BD [4]—has not been fully realised. This is primarily due to the fact that the solution algorithms associated with decomposition techniques typically use MIP solvers as black boxes. As a result, the master and subproblems arising from the application of decomposition are each solved separately. Thus, primal heuristics are applied without any knowledge of the original problem. This has a negative effect on the quality of the solutions found and, hence, the heuristic performance of the BD algorithm. Further, very few, if any, of the many primal heuristics within MIP solvers have been designed with a consideration of decomposition techniques.

The contributions of this paper are:

- The formal presentation of a general enhancement technique derived from the improved use of LNS heuristics within the BD algorithm—the Large neighbourhood Benders' search (LNBS).
- The development of a trust region (TR) heuristic to enhance the BD algorithm by complementing the available heuristics within SCIP.
- A computational study evaluating the improved heuristic performance of BD through the use of the LNBS and TR heuristic.

This paper is structured as follows: An overview of LNS heuristics and their interaction with BD will be presented in Section 2. Previous work on LNS heuristics and trust region methods for BD will be discussed. Section 3 will present the general enhancement approach of the LNBS and a trust region heuristic for use within a branch-and-cut implementation of BD. The implementation of the LNBS and the TR heuristic within the solver SCIP will be described in Section 4. A comprehensive computational study of the proposed methods will be presented in Section 5. Finally, concluding remarks and possible directions for future work will be given in Section 6.

2 Large neighbourhood search and Benders' decomposition

LNS heuristics aim to find improving solutions for a general MIP instance by exploring a restricted set of primal feasible solutions. There are two fundamental aspects of LNS heuristics, the definition of the neighbourhood that is searched for improving solutions and searching this neighbourhood by solving a sub-MIP, described as the auxiliary problem of the LNS heuristic. The neighbourhood is

defined by fixing variables or adding constraints to the auxiliary problem that will hopefully form a problem that is easier to solve than the original MIP. The auxiliary problem is then solved over the restricted feasible region. The objective function of such an auxiliary problem may also be modified to aid the search for improving solutions.

Consider the MIP given by:

$$\min_x \{c^\top x \,|\, Ax \leq b, x \in \mathbb{Z}^p \times \mathbb{R}^{n-p}\}. \tag{1}$$

The set of primal feasible solution for (1) is given by $\mathcal{I} := \{x \,|\, Ax \leq b, x \in \mathbb{Z}^p \times \mathbb{R}^{n-p}\}$. Now consider a polyhedron given by $\mathcal{N}_x$ that defines the neighbourhood for a given LNS heuristic. The auxiliary problem of the LNS heuristic is given by

$$\min_x \{f(x) \,|\, x \in \mathcal{I} \cap \mathcal{N}_x\}, \tag{2}$$

where $f(x)$ is a general function, which need not be linear, that is used to guide the search.

Ideally the design of an LNS heuristics will achieve two main goals: containing high quality solutions in $\mathcal{I} \cap \mathcal{N}_x$ and finding such solutions is not "too" difficult. Attempts to achieve these two goals have lead to the development of a number of LNS heuristics. Examples within the solver SCIP are *Crossover* [29], *DINS* [17], *Local branching* [15], *Proximity search* [16] and *RINS* [13]. For further details about LNS heuristics and those implemented within SCIP, the reader is referred to the PhD thesis of Berthold [6].

2.1 Interaction with Benders' decomposition

While LNS heuristics are a valuable component of MIP solvers, there has been little focus on their interaction with BD. To discuss the use of LNS heuristics with BD, consider the following classical example of a decomposable problem:

$$\min_{x,y} \{c^\top x + d^\top y \,|\, Ax \leq b, Bx + Hy \leq h, x \in \mathbb{Z}^p \times \mathbb{R}^{n-p}, y \in \mathbb{R}^q\}. \tag{3}$$

Problem (3) is commonly described as a two-stage problem, with the first and second stage variables given by x and y respectively. BD exploits the property that for a fixed x solution, the first and second stages become separable.

The application of BD results in the formation of a subproblem, given by

$$z(\hat{x}) = \min_y \{d^\top y \,|\, Hy \leq h - B\hat{x}, y \in \mathbb{R}^q\}. \tag{4}$$

An important observation from (4) is that the feasible region is parameterised by the variables x. More importantly, the dual feasible region of (4), which is given by $\mathcal{U} := \{u \,|\, uH = d, u \leq 0, u \in \mathbb{R}^m\}$, does not depend on the value of x. The BD solution algorithm uses the extreme points and rays from $\mathcal{U}$, denoted by $\mathcal{P}$ and $\mathcal{R}$ respectively, to generate cuts that are added to the master problem.

Exploiting the separability of (3) to form the subproblem leads relaxation of the original problem—by removing the second stage variables and constraints. As a result, a master problem is created, which is initially given by

$$\min_{x,\varphi} \{c^\top x + \varphi \,|\, Ax \leq b, x \in \mathbb{Z}^p \times \mathbb{R}^{n-p}, \varphi \in \mathbb{R}\}, \tag{5}$$

where φ is auxiliary variable that is added as an underestimator of the subproblem objective function value, which can be initially unbounded. Throughout this paper, problems (5) and (4) are labelled the BD master (MP) and subproblem (SP) respectively.

The initial relaxation of MP is formulated with subsets of feasibility and optimality cuts, corresponding to the extreme points and rays in $\bar{\mathcal{P}} \subseteq \mathcal{P}$ and $\bar{\mathcal{R}} \subseteq \mathcal{R}$, both of which can be empty. Given a solution $(\hat{x}, \hat{\varphi})$ to MP, its feasibility and optimality with respect to the second stage constraints is verified by solving SP. In the case that $\hat{x}$ induces an infeasible instance of SP, then the unbounded dual ray $u \in \mathcal{R}$ is used to construct the feasibility cut $u^\top(h - Bx) \leq 0$ and u is appended to $\bar{\mathcal{R}}$. Alternatively, if $\hat{x}$ induces a feasible instance of SP, then an optimal dual extreme point $u \in \mathcal{P}$ is used to form an optimality cut $u^\top(h - Bx) \leq \varphi$. The dual extreme point u is appended to $\bar{\mathcal{P}}$ if $z(\hat{x}) > \hat{\varphi}$. If no feasibility or optimality cut is added to the master problem, then $(\hat{x}, \hat{\varphi})$ is an optimal solution to (5). As a slight abuse of notation, (5) with additional Benders' optimality and feasibility cuts will also be labelled as MP.

Now, consider the MP at an intermediate stage during the solution algorithm. All solutions (x, φ) must satisfy the system of inequalities

$$\begin{aligned} & Ax \leq b, \\ & 0 \geq u^\top(h - Bx) \quad \forall u \in \bar{\mathcal{R}}, \\ & \varphi \geq u^\top(h - Bx) \quad \forall u \in \bar{\mathcal{P}}, \\ & \varphi \geq \underline{\varphi}, \\ & x \in \mathbb{Z}^p \times \mathbb{R}^{n-p}, \\ & \varphi \in \mathbb{R}. \end{aligned} \tag{6}$$

Using the notation from above, the set of solutions satisfying the system of equations (6) is denoted by $\bar{\mathcal{I}}$. The set $\mathcal{I}$ is used to denote all feasible solutions satisfying (6) when $\bar{\mathcal{P}}$ and $\bar{\mathcal{R}}$ is replaced by $\mathcal{P}$ and $\mathcal{R}$. It can be observed that $\mathcal{I} \subseteq \bar{\mathcal{I}}$, a fact that will be used later in this paper. Since this master problem is at an intermediate stage of the solution process, the sets $\bar{\mathcal{P}}$ and $\bar{\mathcal{R}}$ do not provide an optimal underestimation of (4). Thus, the LNS auxiliary problem (2) searches for improving solutions in a feasible region that is a neighbourhood within a relaxation of the original problem, i.e., for MP. As such, there is no guarantee on the feasibility of solutions with respect to the original problem. Also, solving the auxiliary problem may find a solution that improves upon the current best know solution from $\bar{\mathcal{I}}$; however, the solution $\hat{x}$ may not improve upon the best know solution from $\mathcal{I}$.

The LNBS, described in Section 3.1, addresses the issues of feasibility and solution quality by solving the auxiliary problem by BD. The proposed approach solves the auxiliary problem by implicitly considering all second stage constraints. Thus, all solutions found by the auxiliary problem are contained in $\mathcal{I}$, as opposed to $\bar{\mathcal{I}}$. This leads to higher quality solutions found by LNS heuristics when employing BD.

2.2 Related work

Various approaches have been proposed to enhance the BD algorithm through the use of LNS heuristics. These approaches have been in the form of bespoke

algorithms, and there has been no attempt made to systematically integrate LNS heuristics and BD. The first example of a bespoke approach integrating LNS heuristics and BD, in the form of an enhancement technique, is the use of Local Branching [15] to accelerate BD by Rei et al. [28]. The BD algorithm is augmented by phases of Local Branching in an effort to improve the incumbent solution. It is shown by Rei et al. [28] that employing Local Branching significantly improves the convergence of the BD algorithm.

More recently, the Proximity Search heuristic developed by Fischetti and Monaci [16] has been applied in a Benders' decomposition context by Boland et al. [8]. Similar to Proximity Search [16], each iteration of Proximity Benders' [8] sets an upper bound relative to the incumbent solution in an attempt to find an improving solution. A proximity function, such as the Hamming distance, replaces the objective function to focus the search around the current incumbent solution. The results presented by Boland et al. [8] demonstrate that employing BD within the Proximity Search algorithm aids in improving the primal bound more quickly than by the default BD algorithm.

This paper proposes an alternative integration of LNS heuristics and BD compared to the custom-built methods presented above. A major contribution this work is the generalisation of the integration of BD and LNS heuristics—going beyond the custom integration methods of Rei et al. [28] and Boland et al. [8].

2.2.1 Trust region methods

The use of LNS heuristics to enhance BD shares many similarities with trust region methods [10,32]. Briefly, trust region methods are a numerical optimisation approach employing an iterative algorithm that starts from an incumbent solution and progressively steps towards the optimum. In each step, the incumbent solution is updated by finding a trial step from the solution of a trust region subproblem, which is an optimisation problem that can be defined over a neighbourhood of the original feasible region. The algorithm terminates when convergence limits are reached. For further details regarding trust region methods, the reader is referred to Conn et al. [10].

Trust region methods have been previously applied with BD to aid the convergence of the algorithm and reduce the solving time of the master problem [23, 30, 31]. The use of these techniques is motivated by the observed large distances between master problem solutions from consecutive iterations of the BD algorithm. Using a trust region subproblem—solved by BD—to find the next trial step reduces the distance between master problem solutions and focuses the generated cuts on a restricted feasible region.

There have been many examples of using trust region techniques with the BD algorithm. One of the earliest examples is the regularisation decomposition of Ruszczyński [30] that involves the addition of a quadratic regularisation term to the objective function. While effective, the inclusion of the quadratic terms results in the addition of a large number of variables and constraints when linearisation techniques are applied. A trust region method is employed by Linderoth and Wright [20] for a parallel implementation of the L-Shaped method, which is a BD algorithm for two-stage stochastic programs with continuous recourse. Santoso et al. [31] propose a trust region method that involves the addition of a linear constraint in the master problem to impose a limit on

the number of changes in solution between consecutive algorithm iterations. Alternatively, Maher et al. [23] minimise the Hamming distance between the previous and current master problem solutions while imposing an upper bound on the objective function value. Building upon the successes of previous trust region approaches, the TR heuristic is proposed in Section 3.2 to improve the heuristic performance of BD.

3 Enhancing LNS heuristics for BD

The following methods are proposed to enhance LNS heuristics when employing the BD algorithm. To highlight the significance of the proposed methods, the concept of *inner* and *outer* algorithms will be introduced. A inner algorithm is a method that supports a main algorithm, such as separation methods or primal heuristics within a branch-and-bound based solver or the Magnanti-Wong [22] cut strengthening technique for BD. An outer algorithm has a more dominant role by wrapping around the main algorithm and guiding the solution process, such as trust region approaches. Inner and outer algorithms have vastly different implementation approaches and potentially different computational effectiveness.

The previous work on enhancements for BD that exploit heuristic or trust region methods presented above are outer algorithms for BD. For example, Proximity Benders' [8] employs Proximity Search as the main algorithm and BD is used as the solution approach for the auxiliary problem. Similarly, in previous trust region approaches [20,23,30,31] the trust region algorithm is the outer algorithm that imposes a restriction on the master problem and then BD is the inner algorithm that solves the resulting restricted problem.

The following methods described in Sections 3.1 and 3.2 are designed to fit within the branch-and-cut implementation of BD. This implementation treats BD as an inner algorithm of the branch-and-cut algorithm of the underlying MIP solver; however, BD has a dominate role in proving optimality and feasibility of solutions. Since BD is a dominate algorithm in the branch-and-cut implementation, all of the inner algorithms of a MIP solver can be viewed as inner algorithms for BD. The LNBS, presented in Section 3.1, extends previous work on outer trust region and heuristic algorithms by enhancing all available inner LNS heuristic methods. Further, the TR heuristic, described in Section 3.2, is an inner LNS heuristic variant of trust region methods and Proximity Search.

3.1 Large neighbourhood Benders' search

Consider MP at an intermediate stage in the BD solution algorithm, as given in Section 2.1. The feasible region of the corresponding MP is denoted by $\bar{\mathcal{I}}$. The LNS heuristic auxiliary problem, with a feasible region denoted by $\bar{\mathcal{I}}_{\mathcal{N}} := \bar{\mathcal{I}} \cap \mathcal{N}_x$, is given by

$$\min_{x,\varphi} \{ f(x) + \varphi \,|\, (x, \varphi) \in \bar{\mathcal{I}}_{\mathcal{N}} \}, \tag{7}$$

where $f(x)$ is either $c^\top x$ or an alternative objective function that is selected to guide the large neighbourhood search, such as a proximity function [16]. Note that $\bar{\mathcal{I}}_{\mathcal{N}}$ includes the optimality and feasibility cuts that have been previously

generated within the BD algorithm, given by the dual solutions and rays contained in $\bar{\mathcal{P}}$ and $\bar{\mathcal{R}}$ respectively.

Given a solution $\hat{x}$ to (7), the upper bound on the original problem is computed by

$$UB(\hat{x}) = \begin{cases} c^\top \hat{x} + z(\hat{x}) & \text{if SP is feasible,} \\ \infty & \text{otherwise.} \end{cases} \tag{8}$$

Since $\bar{\mathcal{I}}_{\mathcal{N}}$ represents a restriction of the feasible region given by $\bar{\mathcal{I}}$, the second stage constraints, denoted by $Y := \{(x, y) \,|\, Bx + Hy \leq h;\, x \in \mathbb{Z}^p \times \mathbb{R}^{n-p}, y \in \mathbb{R}^q\}$, are still valid for the auxiliary problem. Given solution $\hat{x}$ to (7), it is possible to compute an upper bound for the auxiliary problem when considering the second stage constraints Y by solving SP. Such an upper bound is given by

$$UB_{aux}(\hat{x}) = \begin{cases} f(\hat{x}) + z(\hat{x}) & \text{if SP is feasible,} \\ \infty & \text{otherwise.} \end{cases} \tag{9}$$

Considering (8) and (9), there are two possible methods for solving the auxiliary problem of LNS heuristics. The classical approach is to solve (7) directly without considering the constraints that have been transferred from the original problem to the subproblems. The second stage constraints are only then considered at the termination of the LNS heuristic algorithm when verifying the feasibility or optimality of the candidate solution. This classical approach will be described as solving the auxiliary problem by branch-and-bound. Limitations of the classical approach are that candidate solutions may be infeasible with respect to the second stage constraints, or more commonly, feasible but of poor quality due to a large second stage cost.

The alternative method, the LNBS, is to employ BD to enforce the subproblem constraints in the auxiliary problem. Consider the sets of dual extreme points and rays given by $\hat{\mathcal{P}} \subseteq \mathcal{P} \backslash \bar{\mathcal{P}}$ and $\hat{\mathcal{R}} \subseteq \mathcal{R} \backslash \bar{\mathcal{R}}$, respectively, and denote the set of solutions satisfying the Benders' cuts computed using $\hat{\mathcal{P}}$ and $\hat{\mathcal{R}}$ as $\hat{Y} := \{(x, \varphi) \,|\, \varphi \geq u^\top (h - Bx), \forall u \in \hat{\mathcal{P}};\, 0 \geq u^\top (h - Bx), \forall u \in \hat{\mathcal{R}}\}$. When employing the LNBS, the auxiliary problem of the LNS heuristics is of the form

$$\min_{x,\varphi} \{f(x) + \varphi \,|\, (x, \varphi) \in \bar{\mathcal{I}}_{\mathcal{N}} \cap \hat{Y}\}. \tag{10}$$

Since the sets $\hat{\mathcal{P}}$ and $\hat{\mathcal{R}}$ are prohibitively large, a constraint generation procedure, such as BD, is useful for solving (10). In the following, this alternative method will be described as solving the auxiliary problem by BD.

Solving (10) by BD executes feasibility and optimality verification, by solving SP, during the LNS heuristic algorithm. Therefore, when employing LNBS all feasible solutions to the auxiliary problem satisfy the second stage constraints. This feature is a major advantage of the LNBS compared to the classical use of LNS heuristics. Performing the optimality check during the execution of the LNS heuristic can lead to improved solution quality with respect to the original problem. This is explained by Lemma 1 and Theorem 1.

Remark 1. *All solutions feasible for* (10) *are feasible for the original problem.*

Lemma 1. *If x' and x'' are optimal solutions to* (7) *and* (10) *respectively, then*

$$UB_{aux}(x') \geq UB_{aux}(x'').$$

Proof. The feasible regions of (7) and (10) are $\bar{\mathcal{I}}_{\mathcal{N}}$ and $\bar{\mathcal{I}}_{\mathcal{N}} \cap \hat{Y}$, respectively. Clearly, the feasible region of (10) is a subset of that for (7). Since the objective function of both problems is $f(x) + \varphi$, then $UB_{aux}(x') \geq UB_{aux}(x'')$. □

Theorem 1. *Let $f(x) = c^\top x$ in (7) and (10). If x' and x'' are optimal solutions to (7) and (10) respectively, then*

$$UB(x') \geq UB(x'').$$

Proof. Since $f(x) = c^\top x$, the computation of the upper bound for the original and auxiliary problems, given by (8) and (9) respectively, are identical. Since $UB_{aux}(x) = UB(x)$, using the result from Lemma 1, $UB_{aux}(x'') \leq UB_{aux}(x')$ if and only if $UB(x'') \leq UB(x')$. □

While the assumption on the objective function in Theorem 1 could seem overly restrictive, this situation is commonly observed in practice. Within SCIP, the vast majority of LNS heuristics that are currently available do not modify the objective function coefficients in the auxiliary problem. The most notable exception to this is Proximity Search, where the objective function in the auxiliary problem is replaced with a proximity function. Therefore, in the majority of LNS heuristics, the LNBS will achieve an upper bound at least as good as that achieved when solving the auxiliary problem by branch-and-bound.

3.2 Trust region heuristic

The TR heuristic is an LNS heuristic inspired by trust region approaches employed to solve BD [20,23,30,31] and the MIP heuristics of Local Branching [15] and Proximity Search [16]. The presentation and implementation of this heuristic is to demonstrate the value in revisiting ideas for use within modern applications of the BD algorithm. Different to previous approaches [20,23,30,31], where the trust region approach is an outer algorithm for BD, the proposed heuristic is an LNS heuristic that is embedded within a branch-and-bound solver. As such, this heuristic is called periodically during the branch-and-cut solution process along with the suite of heuristics available within a MIP solver. This significantly changes the focus of the trust region approach since the trade-off between primal and dual improvement must be balanced.

Consider the problem given by (3), which is decomposed by applying BD. The TR heuristic is applicable in cases where $x \in \{0,1\}^r \times \mathbb{Z}^{p-r} \times \mathbb{R}^{n-p-r}$, since the constraints used to define the neighbourhood $\mathcal{N}_x$ rely on the existence of binary variables. The trust region is formed around an incumbent solution, which is denoted by $\bar{x}$, and is bounded by its objective value, denoted by $\bar{z}$. Let $\mathcal{B}$ be the index set of the binary variables and $\mathcal{B}^+$ contain the indices of binary variables that are non-zero in the incumbent solution. The auxiliary problem within the TR heuristic is formed by adding the following constraint to (5),

$$\sum_{i \in \mathcal{B}^+} (1 - x_i) + \sum_{i \in \mathcal{B} \setminus \mathcal{B}^+} x_i \leq \theta, \tag{11}$$

where $\theta \in \mathbb{R}$ is a variable to count the number of changes between $\hat{x}$ and all other feasible solutions. Additionally, an upper bounding constraint of $c^\top x + \varphi \leq \bar{z} - \epsilon$ is added to (5). The setting of ϵ must consider the expected bound change for solutions within the neighbourhood of the current incumbent. It is possible to set ϵ dynamically during the solution process. The objective function for the TR heuristic is modified to $c^\top x + \varphi + M\theta$. The constant M is a large value that penalises the difference between the feasible solutions in the current iteration and the current incumbent solution.

As mentioned previously, the TR heuristic revisits previous ideas for application in a new context. In this respect, the proposed approach draws upon aspects of Proximity Search [16], where the trust region is imposed by using the Hamming distance. However, this method employs the suggestion by Fischetti and Monaci [16], to formulate the objective function with the sum of the master problem objective and the size of the Hamming distance, through the variable θ. The computational performance of this variant of Proximity Search has not been explicitly evaluated in the context of BD. Further, the use of the proposed TR heuristic as an LNS heuristic within the branch-and-bound algorithm has not been previously considered.

4 Implementation

The proposed enhancement techniques are general approaches for BD implemented within a branch-and-cut solver. While general, the LNBS modifies the algorithm behaviour of the LNS heuristics. Thus, a MIP solver that is available in source code is a necessary part of the implementation. This is due to the fact that, to the best of the authors' knowledge, most modern MIP solvers do not provide callbacks that are called from within sub-MIP heuristics, i.e., LNS heuristics. Note that a sub-MIP is the MIP instance of the auxiliary problem from LNS heuristics.

The implementation of the LNBS extends the BD framework available as part of SCIP 6.0 [18]. Additionally, the TR heuristic extends the set of available LNS heuristics within SCIP, making it the first general purpose LNS heuristic designed for BD. In this section, practical implementation details for the LNBS will be presented followed by a description of the TR heuristic implementation.

While the BD algorithm can be implemented within many different solvers, this is not true for the LNBS. Callbacks that are typically used to implement BD in modern branch-and-cut solvers, such as the generic callback with the `CANDIDATE` context in CPLEX are, at present, not called while solving sub-MIPs. Hence, in CPLEX it is only possible to perform the SP check at the termination of the LNS heuristic through the use of callbacks. In this context, only the final solution from a LNS heuristic is passed to the generic callback from which a Benders' cut can be generated. The ability to modify the internal algorithms of the solver SCIP to implement the LNBS makes this general enhancement technique a unique feature of the internal BD framework.

4.1 Large neighbourhood Benders' search

The general BD framework in SCIP 6.0 provides the necessary ground work for the implementation, since it has a tighter integration with the internal al-

gorithms of SCIP than custom built branch-and-cut implementations. Also, extending the general BD framework with the LNBS means that the proposed methods are available to all applications of BD.

The major implementation feature is the modification of the sub-MIP setup process that is performed by SCIP for all LNS heuristics. The creation of a sub-MIP in SCIP involves a *copying* process, where all component algorithms within the solver are copied from the original MIP to the sub-MIP. This includes, but is not limited to, the heuristics, constraint handlers, propagators, separation routines and presolvers. The purpose of this copying process is related to the software design. Since the sub-MIP is a restricted version of the original MIP, all data structures relevant for solving the latter must be made available for the former.

To make the BD algorithm available for solving the auxiliary problem sub-MIP, the corresponding data structures for the BD framework must be *copied* from the original MIP to the sub-MIP. This achieved by extending the copying mechanism within SCIP. The most important components that must be copied are the BD subproblems, the mapping between the master problem and subproblem variables, and the callback functions for solving the subproblems and generating cuts. In SCIP, the callback types `bendersCopyXyz` and `benderscutCopyXyz` have been added to the BD framework to support this copying process. Additionally, the function `SCIPcopyBenders` is added to the SCIP public API to perform the copying of all BD data structures, which is called during the copying process for creating sub-MIPs.

An important feature of the implementation from a software design perspective is the copying of the BD subproblems to the sub-MIP. The implementation of the LNBS avoids copying the BD subproblems by exploiting properties of the LNS auxiliary problem. As explained in Section 3.1, the feasible region $\bar{\mathcal{I}}_{\mathcal{N}}$ is a subset of $\bar{\mathcal{I}}$. As such, all second stage constraints from the original problem are valid for the auxiliary problem and thus, SP is a valid Benders' cut generating LP. A copy of each SP is not necessary when applying BD to solve a sub-MIP, but the original SPs can be used.

Since the LNBS employs BD to solve the auxiliary problem, this process can be very time consuming—especially considering that the auxiliary problem may already be a difficult optimisation problem. Thus, working limits have been imposed to reduce the amount of time that is spent generating Benders' cuts while solving the auxiliary problem. The working limits available in the implementation are:

i) A maximum tree depth at which the LNBS is employed. The maximum depth is based upon the branch-and-bound tree created while solving the master problem. If an LNS heuristic is called at a node depth greater than the maximum depth, then the LNBS is not employed. This parameter is set to either 20 or ∞ for the computational results.

ii) The number of times the subproblems are called when solving the auxiliary problem is limited to 10.

iii) A parameter is provided to limit number times the subproblems are called when solving the LP of the auxiliary problem. In the computational results, this parameter is set to ∞.

4.2 Trust region heuristic

The TR heuristic is an LNS heuristic implemented within the plugin structure of SCIP. The first step of the implemented algorithm is to check whether it is possible execute the TR heuristic. In order for the TR heuristics to be executed there must be i) at least 2 binary variables in the master problem, ii) an incumbent solution, iii) at least one node has been processed since the incumbent was found and iv) the incumbent solution was not found by the trivial heuristic. If these conditions are satisfied, then, similar to other LNS heuristics available within SCIP, a sub-MIP is created as a copy of the original problem. The sub-MIP is then modified as described in Section 3.2.

The resulting sub-MIP is solved, with working limits, to find an improving solution for the original problem. The working limits are an important part of the implementation that helps to ensure the execution of the TR heuristic does not have a negative effect on the overall performance of the BD algorithm. Similar to many LNS heuristics, the working limits that are imposed are related to i) the calling frequency, ii) the number of nodes to process, iii) the number of stalling nodes and iv) the number of best solutions found. For the TR heuristic, the working limits are set as follows:

i) The calling frequency is used to limit how often the heuristic is executed. This is imposed by setting a depth frequency f, so that the heuristic is called at nodes at each f-th depth level. By default, the TR heuristic is disabled by setting f to -1. In the computational experiments when the TR heuristic is enabled, f is set to 25.

ii) To avoid excessive run times for the heuristic, a limit l on the number of nodes to process is imposed. This is given by a function of the number of nodes processed (l_p), solutions found by the TR heuristic (s) and TR heuristic calls (e)

$$l = Qn_p(1 + 2(s + 1)/(e + 1)) - Se + l_o,$$

where $Q \in [0, 1]$ is a factor of the total number of nodes, $S \geq 0$ represents a setup cost in terms of nodes and l_o is a constant offset for the number of nodes to be processed. It is possible that the maximum number of nodes for the auxiliary problem could decrease to zero.

iii) The stall nodes is the number of nodes without incumbent improvement. This is set to $\max\{n/10, 10\}$.

iv) The limit on the number of best solutions found is set to 3.

A feature of the TR heuristic is that if the sub-MIP solves to find an improving solution, then the heuristic is immediately executed again. In this case, the sub-MIP is set up with the solution that was found during the previous execution of the heuristic. The TR heuristic will terminate only when no improving solution is found.

As described in Section 3.2, a set of constraints are added to the sub-MIP and an additional term is added to the objective function. The parameters for these modifications are $\epsilon = 10^{-2}$ and $M = 100$. For working limit ii), in the computational experiments the values of the run time parameters are set to $Q = 0.05$, $S = 100$ and $l_0 = 1000$.

5 Computational experiments

The computational experiments will assess the performance of the LNBS and TR heuristic within the general BD framework of SCIP 6.0. Since the primary focus of this paper is the BD algorithm, the computational results will focus purely of the impact that the proposed approaches have on this algorithm. The results presented in this section will provide insight into the main drivers of improved heuristic performance for the LNBS and TR heuristic.

Two different problem sets will be used for the computational results: the unrooted set covering connected subgraph problem (USCCSP) and the stochastic capacitated facility location problem (SCFLP). These test sets were selected because they contain instances that are unable to be solved to optimality within the given time limit. Since the motivation for developing the proposed methods is to improve the heuristic performance of the BD algorithm, the results will be evaluated using the primal integral [5], which will be described below. In addition, the primal bounds at the end of computation will be reported. The shifted geometric mean (shmean) is used when reporting the aggregation of results across subsets of instances. A shift of 100 is used for the primal integral and 10 for the primal bound.

Figures have been used to aid the visual interpretation of the results. These figures, except for Figure 1, are presented as scatter plots to highlight the overall effect of the evaluated algorithms. All data points in the bottom right portion of the figure (showing better performance of the algorithm on the vertical axis) are coloured red, where the data points in the upper left portion (showing better performance of the algorithm on the horizontal axis) are coloured blue. The data reported in the figures is also presented in tables that are included in Appendix C. All of the analysis in this paper has been performed using IPET [19].

The LNBS is an extension of the BD framework that is available in SCIP 6.0 [18] and the TR heuristic has been added to a development version of SCIP for these computational experiments. All computational experiments have been performed using SoPlex 4.0 as LP solver. The computational experiments have been conducted using a cluster consisting of Intel(R) Xeon(R) CPU E5-2670 2.5GHz processors with 64 GB RAM.

Primal integral. The primal integral is a measure that aims to capture the trade-off between the computational effort and the quality of solutions found. Let $\hat{\delta}$ denote the best known primal bound and $\delta(t)$ the primal bound at time t. The primal gap at time t is given by

$$\gamma(t) = \begin{cases} 1 & \text{if } \delta(t) = \infty \text{ or } \delta(t) \times \hat{\delta} < 0, \\ \dfrac{\delta(t) - \hat{\delta}}{\max\{\delta(t), \hat{\delta}\}} & \text{otherwise.} \end{cases}$$

The primal integral is then given by

$$PI = \int_0^T \gamma(t) \mathrm{d}t,$$

where T is either the run time to solve the instance or the time limit if the instance is unsolved.

Settings used for computational experiments. A number of different settings for SCIP and the BD framework are used in the computational experiments. For all experiments *presolving* and *propagation* are disabled and for the SCFLP instances *separation* is disabled. In addition, the ALNS and RENS heuristics are disabled. Within the BD framework, the generation of cuts on improving, non-optimal integer feasible solutions is enabled. Also, the LNBS max depth is set to 20 and ∞ for the USCCSP and SCFLP instances respectively. All other settings are used as per default for SCIP 6.0.

The settings used for the evaluation of the LNBS and TR heuristic (including the non-default settings stated previously) are:

- *Benders*: the default BD algorithm within SCIP 6.0.
- *LNS check*: *Benders* with the LNBS enabled.
- *Trust Region*: *Benders* with the TR heuristic enabled by setting the frequency to 25.
- *TR LNS check*: *Trust Region* with the LNBS enabled.

A time limit of 3600 seconds has been used for all experiments.

5.1 Unrooted set covering connected subgraph problem

The USCCSP is variant of graph connectivity problems, which was originally proposed by Maher and Murray [24] for the analysis of HIV amino acid sequences. Consider a graph $G = (V, E)$, where V is the set of vertices and E is the set of edges connecting the vertices V. The set P contains features that are observed on the vertices V and P^v denotes the set of features that are observed on vertex v, where $P^v \subseteq P$. The objective of the USCCSP is to identify a minimum cost set of vertices $\bar{V} \subset V$, such that $\cap_{v \in \bar{V}} P^v = P$ and the graph induced by the selected vertices $G(\bar{V}) = (\bar{V}, E(\bar{V}))$ is connected. An alternative formulation to that presented by Maher and Murray [24] has been used in this paper since initial experiments showed it to be more computationally effective. In the interests of space, the problem formulation is only provided in Appendix A.

The instances for the USCCSP have been constructed using set covering problem (SCP) instances from the OR-Library [3]. Specifically, `scp4`{1-10}, `scp6`{1-5}, `scpclr1`{0-3}, `scpcyc`{06-10} (`scpcyc11` was too large and exceeded the memory of the available machine), and `scpe`{1-5}. The remaining SCP instances from the OR-Library formed USCCSP instances that were too easy for the computational experiments and thus did not provide any useful insights into the proposed algorithms.

The instances for the USCCSP are formed by using the constraint matrix structure of the SCP instances. The columns of the SCP instances represent the vertices of the underlying graph and the rows represent the features that must be covered in an optimal solution. Edges for the graph are randomly generated using the random number generator within SCIP as follows: For every vertex pair (i, j), where $i, j \in V$, select a random number α between 0 and 1. If $\alpha < \rho$, where ρ is the probability that two edges are connected, then an edge is added connecting i and j. For the computational experiments $\rho = 0.05$. Using the 29 SCP instances, random seeds of 1, 2, 3, 5 and 7 were used to produce different

USCCSP instances. A total of 145 instances were used in the computational experiments.

5.2 Stochastic Capacitated Facility Location Problem

The SCFLP formulation is adapted from the model developed by Loveaux [21]. The stochastic variant of this problem involves selecting a set of facilities in the first stage and determining the amount of demand from each customer that is serviced by the open facilities in the second stage. The sets of facilities and customers are denoted by I and J respectively. The stochasticity for this problem is in the demands of the customers, which are given by the scenarios in set S. The following formulation is used for the computational experiments:

$$
\begin{aligned}
\min \quad & \sum_{i \in I} f_i x_i + \frac{1}{|S|} \sum_{s \in S} \sum_{i \in I} \sum_{j \in J} q_{ij} y_{ij}^s, & \\
\text{subject to} \quad & \sum_{i \in I} y_{ij}^s \geq \lambda_j^s & \forall j \in J, \forall s \in S, \\
& \sum_{j \in J} y_{ij}^s \leq k_i x_i & \forall i \in I, \forall s \in S, \\
& \sum_{i \in I} k_i x_i \geq \max_{s \in S} \sum_{j \in J} \lambda_j^s, & \\
& x_i \in \{0,1\} & \forall i \in I, \\
& y_{ij}^s \geq 0 & \forall i \in I, \forall j \in J, \forall s \in S.
\end{aligned}
\tag{12}
$$

In the above formulation, the demand of customer j in scenario s is denoted by λ_j^s and and the capacity of facility i is denoted by k_i. The variable x_i equals 1 if facility i is opened, which has a cost of f_i, and 0 otherwise. The variable y_{ij}^s represents the amount of demand of customer j that is served by facility i in scenario s, which has a cost of q_{ij} per unit of demand.

The data for the deterministic capacitated facility location problem has been collected from the CAP instances of the OR-Library [3]. For this paper, the instances with 25 and 50 customers are used. Taking these deterministic instances, the stochastic variant is constructed using the method described by Bodur et al. [7]. Specifically, for each scenario the demand for customer j is sampled from a normal distribution with a mean of λ_j and a standard deviation sampled from a uniform distribution in the range $[0.1\lambda, 0.3\lambda]$. Stochastic instances consisting of 250 scenarios have been generated for these experiments. From the 24 CAP instances, random seeds of 1, 2, 3, 5 and 7 were used to produce different SCFLP instances. A total of 120 instances were used in the computational experiments.

5.3 Solution quality using the LNBS

The theoretical basis for the LNBS is that improved primal bounds from LNS heuristics can be achieved by solving the auxiliary problem by BD. While possible theoretically, MIP solvers are a complex collection of algorithms and working limits may mean that this behaviour is not observed in practice. Thus, the computational study presented in this section evaluates the potential of the LNBS to improve the primal bounds found by LNS heuristics within the solver SCIP.

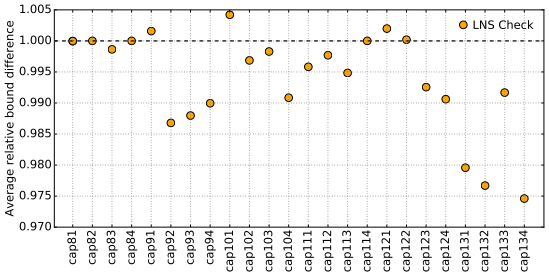


Figure 1: Comparison of bounds achieved by executing RINS by default and with the LNBS for the SCFLP instances using random seed 1.

This experiment focuses on the impact of the LNBS on the primal bounds found by the LNS heuristic RINS when solving the SCFLP instances by BD. Each time RINS is called during the solution process the heuristic is called twice with default working limits: First, the auxiliary problem is solved by branch-and-bound and second, the auxiliary problem is solved by BD. To ensure that the solving behaviour of the second algorithm is not affected by the first, none of the solutions found are added to the master problem, i.e. RINS is executed for information purposes only. Also, the two auxiliary problems are formed prior to either of the algorithms being executed. Finally, when the final solutions from each execution of RINS are checked for feasibility by the BD subproblems, no cuts are generated for the master problem.

Figure 1 presents an evaluation of the solution quality for the LNBS is employed when solving the SCFLP instances generated with a random seed of 1. The presented results are the arithmetic means of the ratio between the best solutions found at each call of the RINS heuristic when the auxiliary problem is solved by branch-and-bound and by BD. For each call of the RINS heuristic, the ratio is only calculated if both algorithms find valid solutions. A ratio less than 1 indicates that the LNBS produces a better primal bound than solving the auxiliary problem by branch-and-bound.

It can be seen in Figure 1 that the LNBS produces a better bound on average. For many of the calls to RINS, the solution found by both algorithms is identical. As such, the average results do not fully show the extent to which the LNBS can aid in improving the primal bound; however, it does provide an overview of the performance. The results show that an improvement of up to 2.5% per call of RINS can be achieved, where the largest improvement for a single call of RINS is 12.5% for `cap134`.

While the LNBS achieves an average improvement in the solution quality over the solving the auxiliary problem by branch-and-bound, there are cases where the opposite in true. In Figure 1, there are four instances, `cap91`, `cap101`, `cap121` and `cap122`, where the average ratio is greater than 1. Also, for an individual call to RINS, the worst performance of the LNBS is observed for instance `cap134` with a solution quality 8.5% worse than compared to solving the auxiliary problem by branch-and-bound. The reason for this behaviour is due to fact that LNS heuristics have many working limits that are used to restrict the run time devoted to solving the auxiliary problem—such as best solution and node limits. These limits can reduce the efficacy of the LNBS, since their use means that the theoretical result may not be realised. However, the results

presented in Figure 1 demonstrate that even with the default working limits that solving the auxiliary problem by BD can improve the quality of solutions found by LNS heuristics.

5.4 Improving heuristic performance using the LNBS

As demonstrated in Section 5.3, solving the auxiliary problem of LNS heuristics by BD can have a significant effect on the quality of the solutions found. The main aim of the LNBS is to improve the overall heuristic performance of the BD algorithm by finding high quality solutions early in the solution process. This effect is observed through a decrease in the primal integral compared to the default BD implementation.

5.4.1 Unrooted set covering connected subgraph problem

The solver performance on the USCCSP instances—in terms of run time, nodes and primal integral—is mixed and thus these instances provide a good test set to demonstrate the potential and limitations of the LNBS. A comparison of the primal integral between *Benders* and *LNS check* is presented in Figure 2. It can be seen in Figure 2 that across the test set the performance of the LNBS varies significantly. There are many instances where *Benders* achieves a superior primal integral; however, there are also many instances where *LNS check* reports an improvement in the primal integral. The main feature dictating the performance of *Benders* and *LNS check* is the difficulty of the instance. The more difficult the instance, the better performance observed from applying the LNBS.

Two different markers are used in Figure 2 to highlight the instances requiring more (dots) and less (crosses) than 1000 nodes when using the *Benders* setting. This threshold on the number of nodes is used to indicate the difficulty of the instances and identify when it is expected that the LNBS will perform well.

The performance of the LNBS on the most difficult instances is further highlighted with shmean of the primal integral. For the instances where both

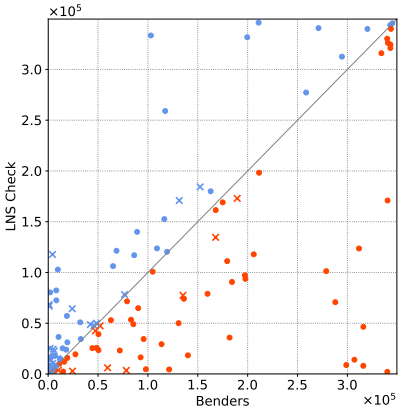


Figure 2: Comparing the primal integral using *Benders* and *LNS check* for the USCCSP instances.

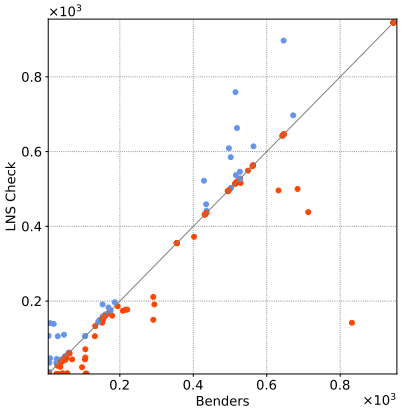


Figure 3: Comparing the primal bound using *Benders* and *LNS check* for the USCCSP instances.

Benders and *LNS check* exceed 1000 branch-and-bound nodes, the shmean of the primal integral is 41 344.2 and 37 278.9 for the two settings respectively. This is a 9.81% decrease in the primal integral when using the LNBS. In comparison to the complete test set, the shmean of the primal integral for *Benders* and *LNS check* is 26 446.1 and 27 993.8 respectively—a 5.9% degradation. Overall, the results demonstrate that for sufficiently difficult instances the LNBS can achieve significant performance improvements in the heuristic behaviour of BD.

The LNBS is expected to improve the quality of the solutions found by LNS heuristics and thus leading to a reduction in the primal bound. This effect can be observed in Figure 3, which compares the objective values from the best solution found between *Benders* and *LNS check*. Importantly, for 97 of the 140 instance *LNS check* finds a solution with a primal bound at least as good as that found by *Benders*. An overall improvement in the primal bounds is achieved by *LNS check* compared to *Benders*. Across the complete test set, the shmean of the primal bound for *Benders* and *LNS check* is 180.203 and 172.633 respectively—a 4.2% reduction.

5.4.2 Stochastic Capacitated Facility Location Problem

The performance of BD, and evaluated algorithms, has much less variation for the SCFLP test set compared to the USCCSP test set. This is primarily due to the fact that the underlying instances, the `cap` instances from the OR-Library, are much more similar to each other than the `scp` instances used for the USCCSP. As such, the results for the LNBS are more consistent for the SCFLP.

A comparison between *Benders* and *LNS check* with respect to the primal integral is presented in Figure 4. None of the SCFLP instances could be solved within the time limit. As such, the ability to quickly find high quality primal bounds is an important feature of the solution algorithm, which is captured by the primal integral. The shmean of the primal integral for *Benders* and *LNS check* is 3955.21 and 3452.0 respectively across the complete test set, corresponding to a decrease of 12.7%.

Typically, LNS heuristics are called periodically during the branch-and-

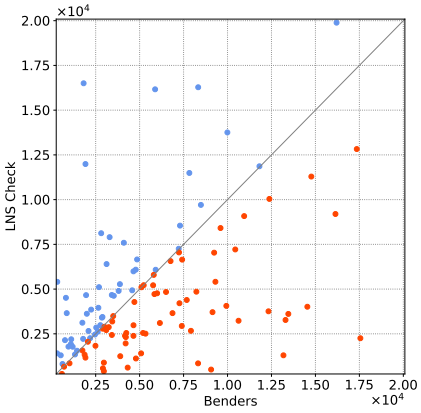


Figure 4: Comparing the primal integral using *Benders* and *LNS check* for the SCFLP instances.

bound search with the frequency of execution in SCIP related to the node depth. For example a frequency of 10 means that the heuristic is called at nodes with a depth of 10, 20, 30, and so on. Considering this algorithmic design for LNS heuristics, a factor contributing to the decrease in the primal integral by *LNS check* with respect to *Benders* is that the minimum number of nodes processed by *Benders* for the SCFLP instances is 4094 and the shmean of nodes processed is 11 084.36. A further important observation is that the shmean of the number of nodes to find the best solution for *Benders* and *LNS check* is 1472.49 and 1217.96 respectively. The large number of nodes to find the best solution means that the LNS heuristics are called more often, thus the LNBS has more opportunities to find higher quality solutions. If the number of nodes to find the best solution is not sufficiently large, then the use of the LNBS will typically degrade the performance of the BD algorithm. In this respect, the SCFLP and difficult USCCSP instances are well suited to the use of the LNBS.

While the LNBS achieves a decrease in the primal integral, a similar improvement in the primal bounds is not observed. In Figure 5, it can be seen

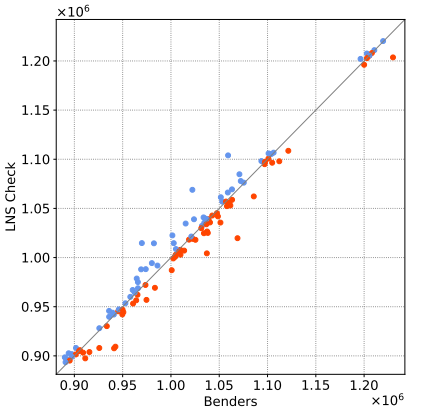


Figure 5: Comparing the primal bound using *Benders* and *LNS check* for the SCFLP instances.

that *LNS check* and *Benders* achieve similar bounds across the complete test set. This result indicates that given sufficient run time, the BD algorithm finds good solutions for the SCFLP instances. However, the speed at which these solutions are found is dependent on the performance of the LNS heuristics. The shmean of the number of nodes to find the best solution decreases by 17.3% when employing the LNBS—highlighting a strength of the proposed enhancement technique. The ability of the LNBS to find good solutions quickly is valuable for problems where finding implementable solutions is time critical.

5.5 Comparing the LNBS and TR heuristic

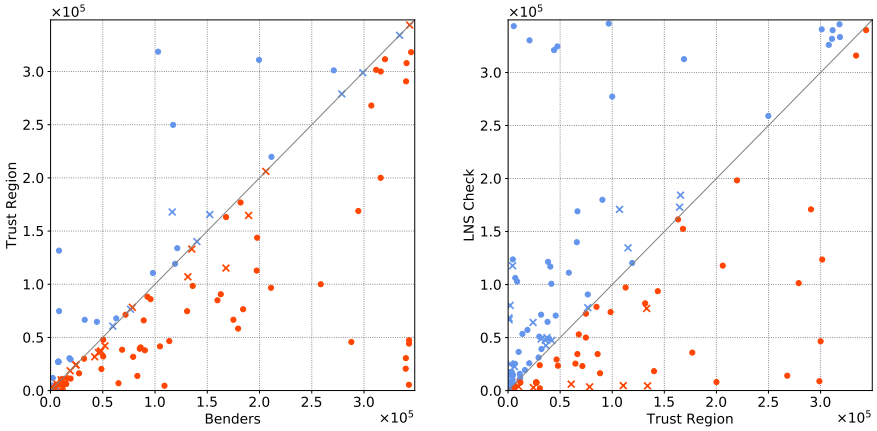
The TR heuristic is proposed as an alternative to the Local Branching [15] and Proximity Search [16] heuristics, the latter which has been used as an outer algorithm for BD by Boland et al. [8]. It must be noted that both Local Branching and Proximity Search are not enabled in SCIP 6.0 by default and thus the default internal BD algorithm does not employ either of these two inner algorithms. While Boland et al. [8] demonstrate the potential of using Proximity Search for enhancing BD, its potential as a component algorithm within a solver has not been evaluated. Similarly, previous variants of the TR heuristic have also been proposed as outer algorithms for BD in the form of trust region methods. Thus, the following computational experiments evaluate the potential of using trust regions as an inner algorithm for BD.

Since the TR heuristic is an LNS heuristic, it can be further enhanced by the use of the LNBS. The computational experiments will first present the improved heuristic performance for BD from the use of the TR heuristic, comparing against *Benders* and *LNS check*. For the SCFLP instances, the improved performance achieved by combining the LNBS with the TR heuristics will be shown.

5.5.1 Unrooted set covering connected subgraph problem

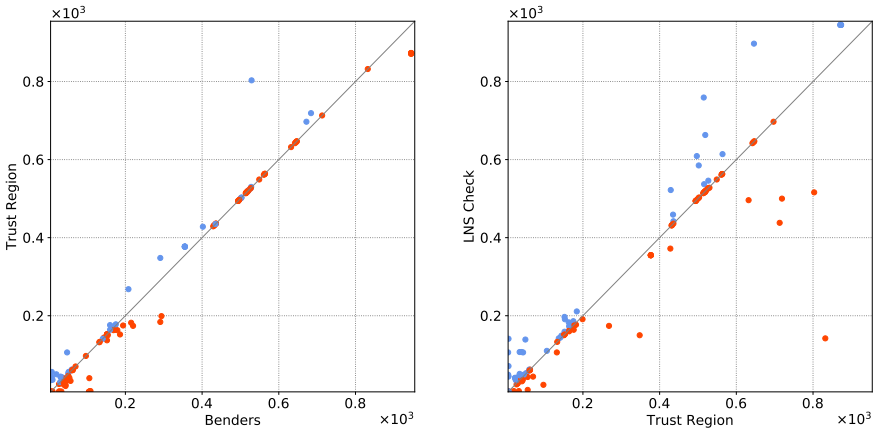
To demonstrate the performance of the trust region heuristic, the primal integral and primal bounds achieved by *Trust Region* will be first compared against *Benders* and then *LNS check*. Figure 6 compares the primal integral for the settings *Benders*, *Trust Region* and *LNS check* and Figure 7 compares the primal bounds. Similar to Figure 2, to highlight the hardness of the instances, those where *Benders* processes less than 1000 nodes are marked with crosses in Figure 6.

The first observation from comparing Figures 6 and 2 is that *Trust Region* is much more stable than *LNS check* when compared to *Benders*. There are many more instances in the lower right portion of Figure 6(a) for *Trust Region*, than when comparing *Benders* and *LNS check*. Specifically, 89 of the 140 instances report an improvement in the primal integral when using *Trust Region* compared to *Benders*. There are also a large number of instances that report a significant decrease in the primal integral when using *Trust Region*—in particular the 5 instances in the lower right corner of the figure. For most of the instances that require less than 1000 nodes (marked with crosses) *Trust Region* achieves a very similar primal integral compared to *Benders*, when *LNS check* typically exhibits an increase. This is due to the frequency setting of 25 for the TR heuristic, meaning that if *Benders* does not produce a tree with a depth



(a) *Benders* and *Trust Region* (b) *Trust Region* and *LNS check*

Figure 6: Comparing the primal integral using *Benders*, *Trust Region* and *LNS check* for the USCCSP instances.



(a) *Benders* and *Trust Region* (b) *Trust Region* and *LNS check*

Figure 7: Comparing the primal bound using *Benders*, *Trust Region* and *LNS check* for the USCCSP instances.

greater than 25 then the TR heuristic may not be called. Many of the easier USCCSP instances are solved at the root node. In these cases, there will be no difference between *Trust Region* and *Benders*. An important result from the computational results is that if the TR heuristic is called, then it generally finds an improving solution—leading to a decrease in the primal integral. This highlights the potency of the TR heuristic for the USCCSP instances.

Comparing *Trust Region* and *LNS check*, Figure 6(b) shows that *Trust Region* is a superior method for improving the heuristic performance of BD. Specifically, across the complete test set, the shmean of the primal integral for *Trust Region* is 20 750.1, which is a 25.9% decrease compared to *LNS check*. This superior performance by *Trust Region* over *LNS check* can be observed by a large number of points in the upper left portion of Figure 6(b). In fact, 92 instances report a better primal integral when using *Trust Region* compared to *LNS check*.

In regards to the primal bound, Figure 7 demonstrates an improvement achieved by *Trust Region* compared to *LNS check*—specifically a 6.1% decrease in the shmean. Considering the *Benders* setting, the relative improvement in the primal integral and primal bound achieved by *Trust Region* is 5.3% and 10% respectively. For the instances where at least 1000 nodes are processed, the improvement in the primal integral over *Benders* by *Trust Region* is 20.6%.

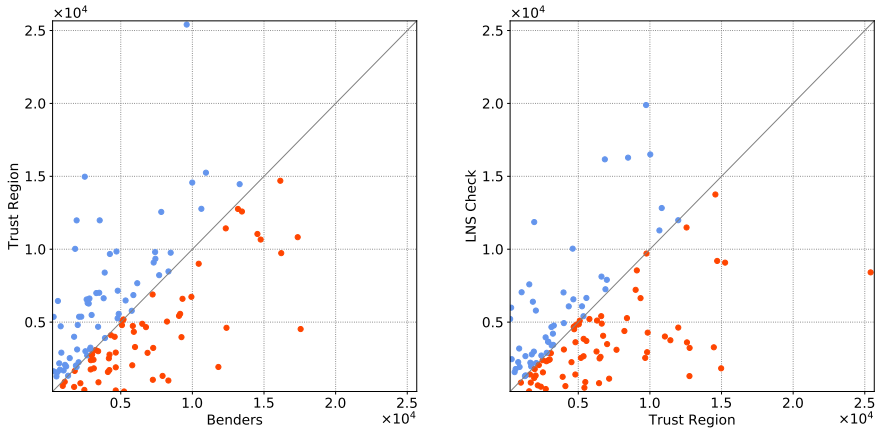
5.5.2 Stochastic Capacitated Facility Location Problem

Figure 8 shows that the performance of *Trust Region* is very inconsistent across the SCFLP test set. Further, *Trust Region* only achieves an improvement in the primal integral over *Benders* for 58 instances—in comparison to 68 for *LNS check*. It can also be observed that *LNS check* achieves a better primal integral than *Trust Region* for 70 of the 120 instances. As a result, *LNS check* and *Benders* outperform *Trust Region* for the SCFLP instances with respect to the primal integral.

These results demonstrate that the problem structure is an important factor driving the performance of the TR heuristic. For *Trust Region*, the TR heuristic was called at most 9 times across the complete test set, and 2.9 times on average. Only on 11 instances the TR heuristic managed to find an integer feasible solution. In comparison, for the USCCSP instances the TR heuristic was called up to 25 times, with average of 4.5 times per instance. When solving the USCCSP instances using *Trust Region*, the TR heuristic manages to find an average of 24.24 integer feasible solution, all of which are improving solutions.

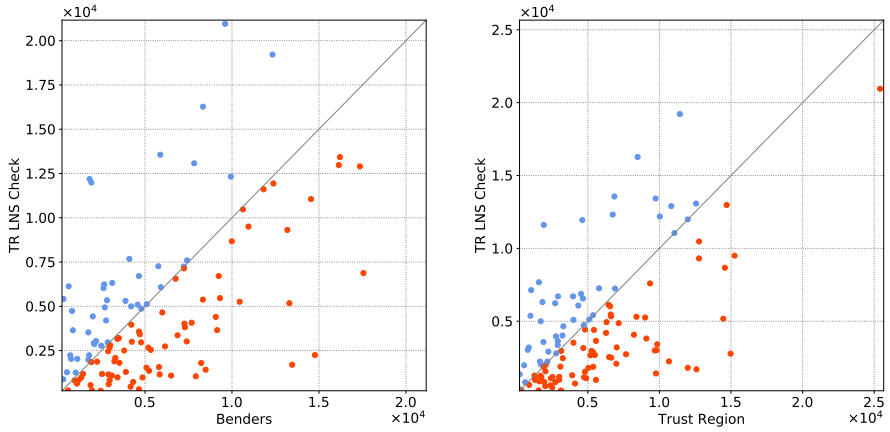
As stated previously, Benders' cuts are generated for all improving, non-optimal, integer feasible solutions. While the TR heuristic failed to find an improving integer feasible solution, checking feasibility with respect to the BD subproblems may have generated additional cuts and, as a result of performance variability, lead to a different solution path. The results presented in Figure 8 show that the inclusion of the TR heuristic has an overall negative effect on the BD algorithm when solving the SCFLP instances.

The primal integral results for *TR LNS check* presented in Figure 9 show that the LNBS significantly improves the performance of the TR heuristic for the SCFLP instances. While the average number of times the TR heuristic is called slightly decreases—from 2.9 to 2.85—improving solutions are now found in at least one call per instance. This result highlights one of the key features

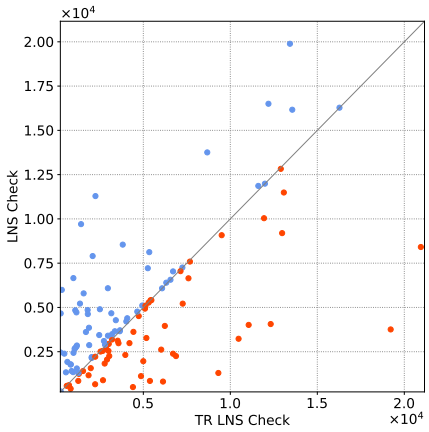


(a) *Benders* and *Trust Region* (b) *Trust Region* and *LNS check*

Figure 8: Comparing the primal integral using *Benders*, *Trust Region* and *LNS check* for the SCFLP instances.



(a) *Benders* and *TR LNS check* (b) *Trust Region* and *TR LNS check*



(c) *TR LNS check* and *LNS check*

Figure 9: Comparing the primal integral using *Benders*, *Trust Region*, *TR LNS check* and *LNS check* for the SCFLP instances.

of the LNBS, where the solution quality is improved by solving the auxiliary problem by BD.

Figures 9(a) and 9(b) also show that there are many instances where *TR LNS check* achieves a lower primal integral than *Benders* and *Trust Region*. Specifically, the shmean of the primal integral for *Benders*, *Trust Region* and *TR LNS check* is 3955.21, 4043.02 and 3130.31 respectively. That corresponds to a respective improvement of 20.9% and 22.6% by *TR LNS check* over *Benders* and *Trust Region*. This improvement is due, in part, to the performance of *LNS check* for the SCFLP instances, since the LNS heuristics that are active in the *Benders* setting are still active with the *TR LNS check* setting. However, Figure 9(c) shows that *TR LNS check* does appear to achieve an improvement in the primal integral over *LNS check*. In regards to the shmean, *TR LNS check* achieves and improvement of 9.3% over *LNS check*. While the TR heuristic alone fails to improve the heuristic performance of BD for the SCFLP instances, the combined use with the LNBS results in a larger number of solutions being found and reduces the primal integral.

6 Concluding remarks

This paper presents two different, but complementary, approaches for improving the heuristic performance of the BD algorithm—the LNBS and the TR heuristic. The main contribution of the LNBS is a systematic way to integrate LNS heuristics and BD as a general algorithmic enhancement technique. Specifically, through a software and algorithmic change to SCIP, the auxiliary problems of LNS heuristics can be solved by BD. The proposed approach extends the integration of these two methods from bespoke approaches previously presented in the literature [8, 28] to all available LNS heuristics within SCIP. The TR heuristic has been developed by building upon the close relationship of LNS heuristics and trust region methods. Employing a trust region approach as an inner algorithm for BD presents an alternative method for using successful trust region algorithms.

A detailed computational study is presented to evaluate the performance of the proposed techniques within the general BD framework available in SCIP 6.0. The initial study shows that the quality of the solutions found by the RINS heuristic improves with the use of the LNBS. These results are shown to translate into an overall improvement to the heuristic performance of BD when solving the USCCSP and SCFLP instances, especially the difficult instances from the USCCSP test set. The TR heuristic is shown to significantly improve the heuristic performance of BD when solving the USCCSP instances. The same performance is not achieve for the SCFLP instances. However, for the SCFLP instances, the performance of the TR heuristic is shown to be greatly improved with the use of the LNBS. A major conclusion from the results is that while the individual strategies of the LNBS and the TR heuristic can greatly improve the heuristic performance of BD, a balanced mix of strategies is crucial for achieving the best performance from the algorithm. The availability of the BD framework within SCIP greatly simplifies the task of developing such balanced strategies.

The proposed approaches and developed software open many possible directions for future research. A main focus of this paper is the extension of previously proposed outer algorithms for use as inner algorithms. Continuing

this investigation is an interesting course of future research; for example using scenario decomposition [1] as a start heuristic for BD. Also, an investigation and analysis of different algorithmic strategies for BD implemented within a branch-and-cut framework will provide great insight for future users of this popular mathematical programming algorithm. Finally, the LNBS is an extension of LNS heuristics, as such the variable fixings and neighbourhood definitions are still made with respect to the master problem. An area of future research will focus on extending neighbourhood definitions to include second stage variables and constraints.

Acknowledgements

This work was support by the Engineering and Physical Sciences Research Council (EPSRC) grant EP/P003060/1. The author would like to thank Ambros Gleixner, Robert Gottwald, Felipe Serrano, Stefan Vigerske and Jakob Witzig for performing code reviews for the general Benders' decomposition framework in SCIP 6.0.

References

[1] S. Ahmed. A scenario decomposition algorithm for 0-1 stochastic programs. *Operations Research Letters*, 41(6):565–569, 2013.

[2] R. K. Ahuja, O. Ergun, J. B. Orlin, and A. P. Punnen. A survey of very large-scale neighborhood search techniques. *Discrete Applied Mathematics*, 123(1-3):75–102, 2002.

[3] J. E. Beasley. A Lagrangian heuristic for set-covering problems. *Naval Research Logistics (NRL)*, 37(1):151–164, 1990.

[4] J. F. Benders. Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik*, 4(1):238–252, 1962.

[5] T. Berthold. Measuring the impact of primal heuristics. *Operations Research Letters*, 41(6):611–614, 2013.

[6] T. Berthold. *Heuristic algorithms in global MINLP solvers*. PhD thesis, 2014.

[7] M. Bodur, S. Dash, O. Günlük, and J. Luedtke. Strengthened Benders cuts for stochastic integer programs with continuous recourse. *INFORMS Journal on Computing*, 29(1):77–91, 2017.

[8] N. Boland, M. Fischetti, M. Monaci, and M. Savelsbergh. Proximity Benders: A decomposition heuristic for stochastic programs. *Journal of Heuristics*, 22(2):181–198, 2016.

[9] S. P. Canto. Application of Benders' decomposition to power plant preventive maintenance scheduling. *European Journal of Operational Research*, 184(2):759–777, 2008.

[10] A. Conn, N. Gould, and P. Toint. *Trust Region Methods*. Society for Industrial and Applied Mathematics, 2000.

[11] J.-F. Cordeau, G. Stojković, F. Soumis, and J. Desrosiers. Benders' decomposition for simultaneous aircraft routing and crew scheduling. *Transportation Science*, 35(4):375–388, 2001.

[12] A. M. Costa. A survey on Benders decomposition applied to fixed-charge network design problems. *Computers & Operations Research*, 32(6):1429–1450, 2005.

[13] E. Danna, E. Rothberg, and C. L. Pape. Exploring relaxation induced neighborhoods to improve MIP solutions. *Mathematical Programming*, 102(1):71–90, 2005.

[14] G. B. Dantzig and P. Wolfe. Decomposition principle for linear programs. *Operations Research*, 8(1):101–111, 1960.

[15] M. Fischetti and A. Lodi. Local branching. *Mathematical Programming*, 98:23–47, 2003.

[16] M. Fischetti and M. Monaci. Proximity search for 0-1 mixed-integer convex programming. *Journal of Heuristics*, 20(6):709–731, 2014.

[17] S. Ghosh. *DINS, a MIP Improvement Heuristic*, pages 310–323. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.

[18] A. Gleixner, M. Bastubbe, L. Eifler, T. Gally, G. Gamrath, R. L. Gottwald, G. Hendel, C. Hojny, T. Koch, M. E. Lübbecke, S. J. Maher, M. Miltenberger, B. Müller, M. E. Pfetsch, C. Puchert, D. Rehfeldt, F. Schlösser, C. Schubert, F. Serrano, Y. Shinano, J. M. Viernickel, M. Walter, F. Wegscheider, J. T. Witt, and J. Witzig. The SCIP Optimization Suite 6.0. Technical Report 18-26, Zuse Institute Berlin, 2018.

[19] G. Hendel. IPET interactive performance evaluation tools. `https://github.com/GregorCH/ipet`.

[20] J. Linderoth and S. Wright. Decomposition algorithms for stochastic programming on a computational grid. *Computational Optimization and Applications*, 24(2-3):207–250, 2003.

[21] F. V. Louveaux. Discrete stochastic location models. *Annals of Operations Research*, 6(2):21–34, 1986.

[22] T. Magnanti and R. Wong. Accelerating Benders' decomposition: Algorithmic enhancement and model selection criteria. *Operations Research*, 29(3):464–484, 1981.

[23] S. J. Maher, G. Desaulniers, and F. Soumis. Recoverable robust single day aircraft maintenance routing problem. *Computers & Operations Research*, 51:130–145, 2014.

[24] S. J. Maher and J. M. Murray. The unrooted set covering connected subgraph problem differentiating between HIV envelope sequences. *European Journal of Operational Research*, 248(2):668–680, 2016.

[25] A. Mercier, J. Cordeau, and F. Soumis. A computational study of Benders' decomposition for the integrated aircraft routing and crew scheduling problem. *Computers & Operations Research*, 32(6):1451–1476, 2005.

[26] F. Oliveira, I. Grossmann, and S. Hamacher. Accelerating Benders stochastic decomposition for the optimization under uncertainty of the petroleum product supply chain. *Computers & Operations Research*, 49:47–58, 2014.

[27] N. Papadakos. Integrated airline scheduling. *Computers & Operations Research*, 36(1):176–195, 2009.

[28] W. Rei, J.-F. Cordeau, M. Gendreau, and P. Soriano. Accelerating Benders' decomposition by local branching. *INFORMS Journal on Computing*, 21(2):333–345, 2009.

[29] E. Rothberg. An evolutionary algorithm for polishing mixed integer programming solutions. *INFORMS Journal on Computing*, 19(4):534–541, 2007.

[30] A. Ruszczyński. A regularized decomposition method for minimizing a sum of polyhedral functions. *Mathematical Programming*, 35(3):309–333, 1986.

[31] T. Santoso, S. Ahmed, M. Goetschalckx, and A. Shapiro. A stochastic programming approach for supply chain network design under uncertainty. *European Journal of Operational Research*, 167(1):96–115, 2005.

[32] Y.-x. Yuan. Recent advances in trust region algorithms. *Mathematical Programming*, 151(1):249–281, 2015.

A Unrooted set covering connected subgraph problem formulation

In Maher and Murray [24], the USCCSP is formulated by integrating the set covering problem with a connected network problem, in the form of a series of shortest path problems. In this paper, an alternative formulation is proposed that significantly reduces the number of variables and constraints in the connected network problem. As opposed to using the shortest path problem to model connectivity, a network flow problem is used. Two additional nodes are added to the network, dummy source and sink nodes, along with the edges (s, j) and (j, t) for all $j \in V$. A flow equal to the number of nodes selected in the set cover is input at the source, and constraints are added to enforce that exactly one edge (s, j), $j \in V$, can have a positive flow. The edge from each of the vertices selected in the set cover to the sink node has a flow capacity of 1, with all other edges connected to the sink node having a capacity of 0. No flow capacity is imposed on the edges in E. Thus, the graph $G(\bar{V})$, induced by the set covering solution $\bar{V}$ is connected if there exists a feasible flow from the source to the sink.

The formulation of the USCCSP used for the computational experiments is given by,

$$\min \quad \sum_{i\in V} c_i x_i + Q\sum_{i\in V} z_i + R\sum_{i\in V}\left(\epsilon_i^+ + \epsilon_i^-\right), \tag{13a}$$

$$\text{subject to} \quad \sum_{i\in V} a_{ip}x_i \geq 1 \qquad \forall p \in P, \tag{13b}$$

$$\sum_{j\in V} y_{sj} = \sum_{i\in V} x_i, \tag{13c}$$

$$y_{sj} \leq |N| w_j \qquad \forall j \in V, \tag{13d}$$

$$y_{sj} \leq |N| x_j \qquad \forall j \in V, \tag{13e}$$

$$w_j \leq w_{j-1} \qquad \forall j \in V\backslash\{0\}, \tag{13f}$$

$$w_j \leq 1 - x_{j-1} \qquad \forall j \in V\backslash\{0\}, \tag{13g}$$

$$\sum_{i\in V} y_{it} \leq x_j \qquad \forall j \in V, \tag{13h}$$

$$\sum_{i\in V} y_{ij} - \sum_{k\in V} y_{jk} = \epsilon_j^+ - \epsilon_j^- \qquad \forall j \in V, \tag{13i}$$

$$\sum_{i\in V} y_{ij} - x_j \leq z_j \qquad \forall j \in V, \tag{13j}$$

$$x_i, w_i \in \{0,1\} \qquad \forall i \in V, \tag{13k}$$

$$y_{ij} \geq 0 \qquad \forall (i,j) \in E, \tag{13l}$$

$$z_i, \epsilon_i^+, \epsilon_i^+ \geq 0, \qquad \forall i \in V. \tag{13m}$$

The objective function minimises the cost of the set cover and any penalties resulting from sending a flow of $|\bar{V}|$ through a disconnected induced graph $G(\bar{V})$. The set of features that must be covered are contained in the set P. Feature p is observed on vertex i if the parameter a_{ip} equals 1. Constraints (13b) ensure that all features are covered by the selected vertices. Connectivity of the induced graph $G(\bar{V})$ is given by the set of constraints (13c)–(13j). The input flow is given by constraint (13c) and the restriction of the flow on a single edge leaving the source is given by the set of constraints (13d)–(13g). Specifically, the bounding constraints (13d)–(13e) and the precedence constraints (13f)–(13g) permits flow only on the vertex with the lowest index that is selected in the set cover. To ensure that flow enters every vertex selected in set cover, the flow into the sink is restricted by constraints (13h). The flow balance in the network is given by constraints (13i). Since $\bar{V}$ may not induce a connected graph, a penalty is imposed for each vertex $v \notin \bar{V}$ that has a non-zero flow passing through it. This is achieved by the constraints (13j). Finally, since it is not guaranteed that G is connected, the variables ϵ_i^+ and ϵ_i^- are include in constraints (13i) to penalise any flow between disconnected vertices.

B Instance sizes

The following tables present the instance sizes of the USCCSP and SCFLP instances. The number of variables and constraints reported are for the master problem and each subproblem. There is only a single subproblem for the

USCCSP instances and 250 subproblems for the SCFLP instances.

ProblemName	Master Vars	Master Cons	Subproblem Vars	Subproblem Cons
scp41	1000	200	56142	7000
scp410	1000	200	56142	7000
scp42	1000	200	56142	7000
scp43	1000	200	56142	7000
scp44	1000	200	56142	7000
scp45	1000	200	56142	7000
scp46	1000	200	56142	7000
scp47	1000	200	56142	7000
scp48	1000	200	56142	7000
scp49	1000	200	56142	7000
scp61	1000	200	56142	7000
scp62	1000	200	56142	7000
scp63	1000	200	56142	7000
scp64	1000	200	56142	7000
scp65	1000	200	56142	7000
scpclr10	210	511	3346	1470
scpclr11	330	1023	7328	2310
scpclr12	495	2047	15096	3465
scpclr13	715	4095	29984	5005
scpcyc06	192	240	2904	1344
scpcyc07	448	672	12644	3136
scpcyc08	1024	1792	58678	7168
scpcyc09	2304	4608	279030	16128
scpcyc10	5120	11520	1341424	35840
scpcyc11	11264	28160	6411332	78848
scpe1	500	50	15426	3500
scpe2	500	50	15426	3500
scpe3	500	50	15426	3500
scpe4	500	50	15426	3500
scpe5	500	50	15426	3500

Table 1: The number of variables and constraints in the master and subproblem for the USCCSP instances using random seed 1.

C Detailed computational results

The following tables present the instance-wise results that corresponds to the results presented in the figures in Section 5. The columns present the primal bound, dual bound, the primal integral, the number of processed nodes, the run time for solving the instance, the number of LP iterations per node and the number of Benders' cuts per node. The final row in each table is the shifted geometric mean for the complete test set. A shift of 100 is used for the primal integral and a shift of 10 for all other columns.

ProblemName	Master Vars	Master Cons	Subproblem Vars	Subproblem Cons
scp41	1000	200	56458	7000
scp410	1000	200	56458	7000
scp42	1000	200	56458	7000
scp43	1000	200	56458	7000
scp44	1000	200	56458	7000
scp45	1000	200	56458	7000
scp46	1000	200	56458	7000
scp47	1000	200	56458	7000
scp48	1000	200	56458	7000
scp49	1000	200	56458	7000
scp61	1000	200	56458	7000
scp62	1000	200	56458	7000
scp63	1000	200	56458	7000
scp64	1000	200	56458	7000
scp65	1000	200	56458	7000
scpclr10	210	511	3542	1470
scpclr11	330	1023	7442	2310
scpclr12	495	2047	15366	3465
scpclr13	715	4095	30256	5005
scpcyc06	192	240	3042	1344
scpcyc07	448	672	12898	3136
scpcyc08	1024	1792	58886	7168
scpcyc09	2304	4608	279490	16128
scpcyc10	5120	11520	1340662	35840
scpcyc11	11264	28160	6408958	78848
scpe1	500	50	15658	3500
scpe2	500	50	15658	3500
scpe3	500	50	15658	3500
scpe4	500	50	15658	3500
scpe5	500	50	15658	3500

Table 2: The number of variables and constraints in the master and subproblem for the USCCSP instances using random seed 2.

ProblemName	Master Vars	Master Cons	Subproblem Vars	Subproblem Cons
scp41	1000	200	56616	7000
scp410	1000	200	56616	7000
scp42	1000	200	56616	7000
scp43	1000	200	56616	7000
scp44	1000	200	56616	7000
scp45	1000	200	56616	7000
scp46	1000	200	56616	7000
scp47	1000	200	56616	7000
scp48	1000	200	56616	7000
scp49	1000	200	56616	7000
scp61	1000	200	56616	7000
scp62	1000	200	56616	7000
scp63	1000	200	56616	7000
scp64	1000	200	56616	7000
scp65	1000	200	56616	7000
scpclr10	210	511	3576	1470
scpclr11	330	1023	7670	2310
scpclr12	495	2047	15510	3465
scpclr13	715	4095	30226	5005
scpcyc06	192	240	3108	1344
scpcyc07	448	672	13020	3136
scpcyc08	1024	1792	59120	7168
scpcyc09	2304	4608	279456	16128
scpcyc10	5120	11520	1340402	35840
scpcyc11	11264	28160	6414710	78848
scpe1	500	50	15788	3500
scpe2	500	50	15788	3500
scpe3	500	50	15788	3500
scpe4	500	50	15788	3500
scpe5	500	50	15788	3500

Table 3: The number of variables and constraints in the master and subproblem for the USCCSP instances using random seed 3.

ProblemName	Master Vars	Master Cons	Subproblem Vars	Subproblem Cons
scp41	1000	200	55872	7000
scp410	1000	200	55872	7000
scp42	1000	200	55872	7000
scp43	1000	200	55872	7000
scp44	1000	200	55872	7000
scp45	1000	200	55872	7000
scp46	1000	200	55872	7000
scp47	1000	200	55872	7000
scp48	1000	200	55872	7000
scp49	1000	200	55872	7000
scp61	1000	200	55872	7000
scp62	1000	200	55872	7000
scp63	1000	200	55872	7000
scp64	1000	200	55872	7000
scp65	1000	200	55872	7000
scpclr10	210	511	3404	1470
scpclr11	330	1023	7316	2310
scpclr12	495	2047	15220	3465
scpclr13	715	4095	29878	5005
scpcyc06	192	240	2992	1344
scpcyc07	448	672	12682	3136
scpcyc08	1024	1792	58334	7168
scpcyc09	2304	4608	278784	16128
scpcyc10	5120	11520	1338936	35840
scpcyc11	11264	28160	6411340	78848
scpe1	500	50	15490	3500
scpe2	500	50	15490	3500
scpe3	500	50	15490	3500
scpe4	500	50	15490	3500
scpe5	500	50	15490	3500

Table 4: The number of variables and constraints in the master and subproblem for the USCCSP instances using random seed 5.

ProblemName	Master Vars	Master Cons	Subproblem Vars	Subproblem Cons
scp41	1000	200	55912	7000
scp410	1000	200	55912	7000
scp42	1000	200	55912	7000
scp43	1000	200	55912	7000
scp44	1000	200	55912	7000
scp45	1000	200	55912	7000
scp46	1000	200	55912	7000
scp47	1000	200	55912	7000
scp48	1000	200	55912	7000
scp49	1000	200	55912	7000
scp61	1000	200	55912	7000
scp62	1000	200	55912	7000
scp63	1000	200	55912	7000
scp64	1000	200	55912	7000
scp65	1000	200	55912	7000
scpclr10	210	511	3460	1470
scpclr11	330	1023	7364	2310
scpclr12	495	2047	15132	3465
scpclr13	715	4095	29768	5005
scpcyc06	192	240	2952	1344
scpcyc07	448	672	12540	3136
scpcyc08	1024	1792	58468	7168
scpcyc09	2304	4608	278942	16128
scpcyc10	5120	11520	1341044	35840
scpcyc11	11264	28160	6407838	78848
scpe1	500	50	15366	3500
scpe2	500	50	15366	3500
scpe3	500	50	15366	3500
scpe4	500	50	15366	3500
scpe5	500	50	15366	3500

Table 5: The number of variables and constraints in the master and subproblem for the USCCSP instances using random seed 7.

	Master Vars	Master Cons	Subproblem Vars	Subproblem Cons
cap101	25	1	1250	75
cap102	25	1	1250	75
cap103	25	1	1250	75
cap104	25	1	1250	75
cap111	50	1	2500	100
cap112	50	1	2500	100
cap113	50	1	2500	100
cap114	50	1	2500	100
cap121	50	1	2500	100
cap122	50	1	2500	100
cap123	50	1	2500	100
cap124	50	1	2500	100
cap131	50	1	2500	100
cap132	50	1	2500	100
cap133	50	1	2500	100
cap134	50	1	2500	100
cap81	25	1	1250	75
cap82	25	1	1250	75
cap83	25	1	1250	75
cap84	25	1	1250	75
cap91	25	1	1250	75
cap92	25	1	1250	75
cap93	25	1	1250	75
cap94	25	1	1250	75

Table 6: The number of variables and constraints in the master and each subproblem for the SCFLP instances using all random seeds. NOTE: there are 250 subproblem for each instance.

	Primal bound	Dual bound	Primal integral	Nodes	Time (sec)	Iter per node	BC per node
scp41	429.00	429.00	1047.42	1	31.010	858.00	3
scp410	529.00	-598573.82	121274.11	1218	3600.250	705.96	1.02
scp42	517.00	517.00	2370.57	192	50.300	19.21	0.12
scp43	516.00	516.00	811.41	1	12.560	222.00	2
scp44	502.00	502.00	18790.04	584	763.480	237.02	0.851
scp45	515.00	515.00	1131.72	1	39.240	775.00	2
scp46	563.00	563.00	1413.55	188	42.360	30.29	0.41
scp47	713.00	-99321.51	140099.71	936	3600.010	5008.67	1.16
scp48	497.00	497.00	880.83	47	27.130	39.09	0.149
scp49	644.00	644.00	1943.00	229	76.440	47.79	0.201
scp61	144.00	144.00	10435.89	24875	3320.180	497.27	0.0341
scp62	153.00	149.01	65053.23	25789	3600.000	67.10	0.121
scp63	194.00	148.38	90270.70	7661	3600.000	253.31	0.592
scp64	152.00	132.00	48719.87	1676	3600.000	7448.57	1.08
scp65	166.00	166.00	11213.74	36989	3043.230	118.19	0.0522
scpclr10	37.00	21.24	92722.09	12286	3600.010	2144.38	0.992
scpclr11	38.00	17.97	136061.56	7012	3600.010	2107.42	0.244
scpclr12	35.00	17.26	8206.43	7205	3600.010	1033.01	0.073
scpclr13	50.00	14.46	152337.78	607	3600.000	2273.99	0.442
scpcyc06	62.00	54.90	16170.04	1171907	3600.020	66.99	7.68e-05
scpcyc07	152.00	117.86	76.25	67227	3600.010	489.21	1.49e-05
scpcyc08	355.00	257.81	7408.47	3039	3600.010	3164.29	0.000329
scpcyc09	945.00	576.50	52081.82	302	3600.000	5926.14	0.00331
scpcyc10	5120.00	—	317360.00	1	3613.350	0.00	0
scpe1	106.00	4.50	339879.61	53373	3600.010	9.05	0.384
scpe2	33.00	4.09	315923.46	35160	3600.010	12.88	0.463
scpe3	105.00	4.35	343061.95	50452	3600.150	12.69	0.4
scpe4	70.00	-49990.10	206077.13	1	3600.000	4627499.00	2.53e+03
scpe5	105.00	4.40	342930.93	57264	3600.000	8.40	0.364
shmean	217.91	—	22374.99	1661	1176.549	360.22	2.75

Table 7: Detailed results for the USCCSP instances with random seed 1 using *Benders*.

	Primal bound	Dual bound	Primal integral	Nodes	Time (sec)	Iter per node	BC per node
scp41	432.00	432.00	2772.07	115	76.390	19.10	0.296
scp410	514.00	514.00	1018.26	1	14.110	167.00	2
scp42	514.00	514.00	1602.79	1	53.270	1355.00	14
scp43	518.00	518.00	24404.15	1	807.370	92646.00	436
scp44	494.00	494.00	695.37	1	15.060	219.00	1
scp45	519.00	519.00	2015.44	1814	99.190	11.23	0.0287
scp46	563.00	563.00	2488.90	349	87.420	52.07	0.421
scp47	436.00	436.00	4602.59	594	134.500	18.96	0.101
scp48	501.00	501.00	2169.16	7832	169.050	69.98	0.0211
scp49	672.00	-2696118.74	19253.97	14396	3600.010	67.96	0.0992
scp61	832.00	-897910.04	298866.86	824	3600.000	1057.25	1.16
scp62	154.00	147.81	95713.01	17838	3600.010	392.29	0.0889
scp63	152.00	152.00	14455.83	33522	1049.310	40.71	0.0264
scp64	141.00	133.01	181846.38	22244	3600.000	209.58	0.0818
scp65	215.00	161.00	85393.47	2642	3600.060	1309.86	1.59
scpclr10	34.00	21.19	109064.19	10632	3600.010	2613.44	1.19
scpclr11	36.00	17.14	113658.41	2267	3600.010	5618.19	0.907
scpclr12	41.00	17.24	86246.49	7743	3600.000	958.60	0.164
scpclr13	51.00	14.45	189523.48	560	3600.000	2670.18	0.579
scpcyc06	63.00	52.75	19242.98	255037	3600.000	489.60	0.000463
scpcyc07	152.00	117.86	69.64	67093	3600.000	489.51	1.49e-05
scpcyc08	355.00	257.81	7515.11	3054	3600.000	3161.78	0.000327
scpcyc09	945.00	576.50	48327.91	306	3600.000	6012.28	0.00327
scpcyc10	5120.00	—	315020.00	1	3611.360	0.00	0
scpe1	28.00	4.58	294528.00	19104	3600.020	31.94	0.958
scpe2	6.00	4.62	271026.39	62007	3600.010	12.59	0.336
scpe3	106.00	4.42	258584.09	55621	3600.000	8.76	0.379
scpe4	109.00	4.40	340247.51	54136	3600.030	8.90	0.381
scpe5	27.00	4.05	307010.56	21949	3600.010	25.94	0.643
shmean	205.66	—	26071.62	2127	1115.586	243.07	2.24

Table 8: Detailed results for the USCCSP instances with random seed 2 using *Benders*.

	Primal bound	Dual bound	Primal integral	Nodes	Time (sec)	Iter per node	BC per node
scp41	549.00	-199146.91	76543.00	617	3600.010	1337.91	1.65
scp410	519.00	519.00	1980.81	1	59.860	858.00	43
scp42	518.00	518.00	3834.27	6722	176.740	37.08	0.0461
scp43	519.00	-99417.63	2803.59	335	3600.000	8904.44	1.81
scp44	684.00	-199320.88	97962.91	1133	3600.010	981.00	1.32
scp45	521.00	521.00	2081.77	3077	72.950	12.73	0.024
scp46	561.00	561.00	1270.02	1	46.440	259.00	8
scp47	647.00	-398475.19	119199.42	962	3600.000	775.18	1.35
scp48	503.00	503.00	6152.05	10217	546.570	102.39	0.0384
scp49	646.00	646.00	10164.21	732	530.390	112.24	0.342
scp61	220.00	137.00	130715.04	2529	3600.000	267.07	1.52
scp62	294.00	145.25	197722.25	2719	3600.010	160.06	1.34
scp63	186.00	146.38	184292.49	4123	3600.010	213.91	1.12
scp64	133.00	133.00	59581.84	449	861.420	200.63	0.572
scp65	173.00	162.01	50421.90	16324	3600.000	262.25	0.161
scpclr10	31.00	21.68	50412.55	12042	3600.010	2766.71	0.339
scpclr11	27.00	18.07	89150.77	14235	3600.000	1296.21	0.0894
scpclr12	44.00	17.11	83091.13	4894	3600.010	1299.49	0.22
scpclr13	52.00	14.46	135193.22	581	3600.010	2496.67	0.448
scpcyc06	60.00	53.98	2199.44	1965838	3600.000	39.16	4.12e-05
scpcyc07	152.00	117.85	72.28	66680	3600.000	490.40	1.5e-05
scpcyc08	355.00	257.81	7471.85	3059	3600.000	3159.57	0.000327
scpcyc09	945.00	576.50	45969.74	308	3600.000	6053.05	0.00325
scpcyc10	5120.00	—	314880.00	1	3606.980	0.00	0
scpe1	97.00	-399968.00	334066.55	1	3600.000	87203.00	1.51e+04
scpe2	107.00	4.35	340007.50	50743	3600.000	13.27	0.456
scpe3	44.00	4.56	311610.15	26783	3600.110	33.16	0.851
scpe4	48.00	4.50	117317.08	20772	3600.020	30.48	0.948
scpe5	57.00	4.16	315986.47	71783	3600.000	10.95	0.219
shmean	221.49	—	33416.57	2235	1799.665	343.67	4.61

Table 9: Detailed results for the USCCSP instances with random seed 3 using *Benders*.

	Primal bound	Dual bound	Primal integral	Nodes	Time (sec)	Iter per node	BC per node
scp41	431.00	431.00	1982.72	111	113.280	18.72	0.64
scp410	516.00	516.00	2664.55	1	282.130	2109.00	120
scp42	528.00	517.00	27331.85	5632	3600.010	264.83	0.555
scp43	525.00	525.00	11895.59	5794	1275.880	166.12	0.189
scp44	632.00	-98795.49	78314.62	423	3600.000	8286.00	2.17
scp45	517.00	517.00	2309.63	29	108.910	53.24	1.14
scp46	564.00	564.00	24113.45	1	297.910	1707.00	83
scp47	436.00	436.00	32160.52	3505	2052.990	1220.30	0.24
scp48	494.00	494.00	4047.71	36	81.220	40.97	0.389
scp49	646.00	646.00	9705.23	2405	1196.790	1327.26	0.249
scp61	173.00	138.01	62911.01	9784	3600.330	332.69	0.202
scp62	179.00	147.06	79083.03	4818	3600.010	371.99	0.837
scp63	402.00	-99783.42	211414.06	2346	3600.000	382.10	0.504
scp64	160.00	131.03	197271.74	11456	3600.030	129.90	0.373
scp65	175.00	162.06	32727.04	5080	3600.010	1300.48	0.475
scpclr10	30.00	21.29	44366.81	23201	3600.010	1321.64	0.369
scpclr11	31.00	17.95	8131.99	7963	3600.000	1720.30	0.106
scpclr12	40.00	17.29	162906.44	7548	3600.010	956.78	0.13
scpclr13	51.00	14.44	131408.39	640	3600.070	2345.65	0.33
scpcyc06	61.00	52.65	11815.14	240489	3600.000	536.96	0.000445
scpcyc07	153.00	117.22	6781.84	65271	3600.000	451.36	4.6e-05
scpcyc08	355.00	257.86	7655.93	3064	3600.000	3162.26	0.000326
scpcyc09	945.00	576.50	42292.49	312	3600.000	6047.60	0.00321
scpcyc10	5120.00	—	315510.00	1	3610.570	0.00	0
scpe1	8.00	4.53	102949.70	51634	3600.000	26.70	0.259
scpe2	106.00	4.38	343123.51	57149	3600.000	8.50	0.356
scpe3	132.00	-99957.67	343673.17	1	3600.010	2788293.00	4.21e+03
scpe4	10.00	4.51	210999.72	60953	3600.000	13.66	0.289
scpe5	7.00	4.89	68515.77	178981	3600.000	11.14	0.0535
shmean	193.55	—	35412.79	2042	1922.047	467.99	5.01

Table 10: Detailed results for the USCCSP instances with random seed 5 using *Benders*.

	Primal bound	Dual bound	Primal integral	Nodes	Time (sec)	Iter per node	BC per node
scp41	432.00	432.00	2015.32	1	38.330	786.00	12
scp410	521.00	521.00	1076.22	1426	70.110	13.75	0.054
scp42	515.00	515.00	4353.18	1	137.120	2470.00	25
scp43	527.00	527.00	3218.09	13789	644.360	108.66	0.0316
scp44	494.00	494.00	1127.86	1	21.780	218.00	2
scp45	527.00	527.00	71786.36	2873	2072.580	1896.50	0.215
scp46	563.00	563.00	1888.78	196	69.770	26.80	0.107
scp47	435.00	435.00	5011.68	517	134.200	98.91	0.17
scp48	497.00	497.00	6134.86	209	309.420	110.23	0.526
scp49	642.00	642.00	1776.82	1	477.370	40941.00	312
scp61	208.00	-99818.75	116460.88	679	3600.010	1451.30	1.77
scp62	169.00	145.99	179478.47	3830	3600.010	287.57	1.26
scp63	160.00	149.02	18106.65	26551	3600.000	184.19	0.101
scp64	291.00	130.38	278733.42	14	3600.000	384201.43	56.6
scp65	291.00	161.01	159754.70	2259	3600.000	183.29	1.57
scpclr10	39.00	21.48	104761.67	14232	3600.010	1766.51	1.12
scpclr11	44.00	17.72	174998.74	4884	3600.000	2280.37	0.383
scpclr12	41.00	17.25	167984.51	7542	3600.010	973.24	0.18
scpclr13	54.00	14.45	167888.65	773	3600.000	2069.87	0.653
scpcyc06	62.00	54.70	15012.86	1035526	3600.010	69.75	0.000109
scpcyc07	152.00	117.86	76.25	67401	3600.010	488.93	1.48e-05
scpcyc08	355.00	257.86	7460.54	3064	3600.010	3162.33	0.000326
scpcyc09	945.00	576.50	47415.19	306	3600.010	6050.68	0.00327
scpcyc10	5120.00	—	316860.00	1	3608.360	0.00	0
scpe1	10.00	4.53	340669.02	42611	3600.130	15.51	0.544
scpe2	31.00	4.41	320067.49	38552	3600.080	14.76	0.534
scpe3	6.00	4.61	287861.42	30559	3600.010	26.78	0.65
scpe4	20.00	4.38	345067.33	45636	3600.090	8.99	0.492
scpe5	8.00	4.58	199525.75	81385	3600.040	10.60	0.207
shmean	179.70	—	28760.28	1684	1283.258	353.03	3.45

Table 11: Detailed results for the USCCSP instances with random seed 7 using *Benders*.

	Primal bound	Dual bound	Primal integral	Nodes	Time (sec)	Iter per node	BC per node
scp41	522.00	-199508.25	66830.41	377	3600.000	8495.15	1.21
scp410	516.00	516.00	4483.33	1	69.420	685.00	11
scp42	517.00	517.00	4098.33	1	63.090	1149.00	29
scp43	537.00	-99395.99	16570.46	397	3600.000	8446.88	1.59
scp44	585.00	-99450.08	57204.09	2066	3600.000	727.52	0.762
scp45	515.00	515.00	2719.98	1	42.360	691.00	5
scp46	563.00	563.00	3985.43	1326	127.300	245.10	0.116
scp47	438.00	438.00	18382.37	16068	889.120	171.78	0.022
scp48	609.00	-99402.99	68539.62	452	3600.010	4789.56	1.69
scp49	644.00	644.00	4205.42	30	71.940	44.80	0.3
scp61	149.00	140.01	36513.26	12808	3600.000	265.36	0.29
scp62	191.00	146.37	106242.67	4132	3600.000	171.94	1.44
scp63	186.00	147.50	64845.43	2989	3600.010	558.24	1.4
scp64	142.00	132.01	25786.09	3191	3600.010	1897.45	0.451
scp65	166.00	166.00	10079.88	75211	3579.140	52.85	0.0218
scpclr10	32.00	21.29	16385.01	13898	3600.000	1839.10	0.85
scpclr11	24.00	18.20	74065.77	27771	3600.000	658.26	0.056
scpclr12	41.00	17.21	72549.82	7905	3600.000	939.44	0.183
scpclr13	51.00	14.46	184318.65	709	3600.000	2209.85	0.48
scpcyc06	62.00	53.16	12079.14	421553	3600.090	286.13	0.000159
scpcyc07	152.00	117.86	94.49	67376	3600.010	488.98	1.48e-05
scpcyc08	355.00	257.81	7684.45	3047	3600.010	3164.25	0.000328
scpcyc09	945.00	576.50	47395.13	309	3600.000	6046.30	0.00324
scpcyc10	5120.00	—	317490.00	1	3606.720	0.00	0
scpe1	71.00	4.54	330292.96	48715	3607.140	13.27	0.449
scpe2	5.00	5.00	8078.30	51925	224.070	16.78	0.0161
scpe3	44.00	4.44	321098.13	18045	3606.950	36.01	1.02
scpe4	44.00	4.52	117859.56	20292	3600.100	32.00	0.949
scpe5	106.00	4.42	343677.92	54211	3600.020	8.87	0.393
shmean	197.99	—	27789.12	2452	1615.974	385.64	1.4

Table 12: Detailed results for the USCCSP instances with random seed 1 using *LNS check*.

	Primal bound	Dual bound	Primal integral	Nodes	Time (sec)	Iter per node	BC per node
scp41	432.00	432.00	14001.37	195	398.370	223.53	0.856
scp410	514.00	514.00	4196.73	1	83.500	902.00	7
scp42	514.00	514.00	3700.70	1	80.540	1746.00	27
scp43	518.00	518.00	2856.06	1	43.240	797.00	11
scp44	494.00	494.00	5254.49	2	299.120	6331.50	51
scp45	663.00	-299126.91	80400.99	379	3600.010	4494.44	1.7
scp46	563.00	563.00	2695.15	2	55.440	430.00	10.5
scp47	442.00	-99545.11	10442.19	390	3600.000	15103.53	1.45
scp48	501.00	501.00	4131.69	2060	96.720	15.02	0.0505
scp49	697.00	-99892.00	31203.37	3296	3600.010	166.08	0.379
scp61	142.00	142.00	8840.90	2416	201.890	30.99	0.0534
scp62	150.00	150.00	34472.86	4535	2063.400	740.90	0.494
scp63	152.00	148.01	25290.93	20315	3600.000	405.85	0.0966
scp64	144.00	133.50	35872.36	14684	3600.010	312.09	0.243
scp65	177.00	162.01	49045.99	9681	3600.010	363.99	0.37
scpclr10	41.00	21.64	123715.50	8462	3600.000	3535.68	0.451
scpclr11	27.00	18.14	29332.33	33169	3600.000	412.59	0.0793
scpclr12	44.00	17.05	116979.00	7772	3600.010	867.31	0.288
scpclr13	53.00	14.42	172976.76	773	3600.010	1918.70	0.495
scpcyc06	60.00	53.05	15849.34	202434	3600.010	522.81	0.00177
scpcyc07	152.00	117.86	82.86	67315	3600.000	489.09	1.49e-05
scpcyc08	355.00	257.81	7502.14	3042	3600.010	3165.44	0.000329
scpcyc09	945.00	576.50	50010.85	300	3600.080	5868.94	0.00333
scpcyc10	5120.00	—	315080.00	1	3612.860	0.00	0
scpe1	45.00	4.63	312574.70	22524	3600.100	35.23	0.887
scpe2	107.00	4.35	340650.80	45060	3600.000	9.75	0.481
scpe3	107.00	4.36	277243.80	48350	3600.070	10.61	0.501
scpe4	6.00	4.82	170911.46	78045	3600.020	18.70	0.183
scpe5	5.00	5.00	14083.82	42097	332.820	17.06	0.033
shmean	190.08	—	24759.25	1836	1296.922	370.97	2.29

Table 13: Detailed results for the USCCSP instances with random seed 2 using *LNS check*.

	Primal bound	Dual bound	Primal integral	Nodes	Time (sec)	Iter per node	BC per node
scp41	549.00	-199441.05	78266.71	394	3600.000	8715.88	1.37
scp410	519.00	519.00	4201.58	1	69.740	708.00	25
scp42	518.00	518.00	17727.01	1185	445.300	252.46	0.316
scp43	517.00	517.00	2956.97	1	65.060	1799.00	13
scp44	500.00	500.00	4706.74	386	93.050	16.73	0.293
scp45	521.00	521.00	15543.72	2369	572.830	40.76	0.164
scp46	561.00	561.00	3058.25	1	44.960	727.00	7
scp47	647.00	-99169.71	120397.50	1250	3600.000	2599.34	0.828
scp48	503.00	503.00	3693.68	11549	474.100	167.10	0.0268
scp49	646.00	646.00	3771.26	520	79.020	12.68	0.0577
scp61	177.00	139.02	50048.29	3655	3600.030	320.91	1.46
scp62	191.00	148.54	93744.84	15123	3600.010	210.35	0.228
scp63	197.00	147.01	90652.41	4748	3600.070	390.21	0.898
scp64	133.00	133.00	6117.12	395	115.820	51.23	0.251
scp65	174.00	162.03	23481.96	20732	3600.010	238.10	0.158
scpclr10	29.00	22.28	39228.13	85751	3600.010	436.61	0.0359
scpclr11	35.00	17.66	139967.80	7872	3600.000	1925.48	0.0845
scpclr12	40.00	17.11	53471.48	4080	3600.000	2055.04	0.0721
scpclr13	43.00	14.43	77507.64	528	3600.000	2138.46	0.455
scpcyc06	61.00	53.11	7640.98	348176	3600.000	340.58	0.000388
scpcyc07	152.00	117.86	84.94	67384	3600.000	488.97	1.48e-05
scpcyc08	355.00	257.81	7641.55	3060	3600.000	3159.00	0.000327
scpcyc09	945.00	576.50	47645.59	306	3600.030	6050.68	0.00327
scpcyc10	5120.00	—	314920.00	1	3609.080	0.00	0
scpe1	23.00	4.57	315951.98	39075	3600.010	24.54	0.633
scpe2	6.00	5.48	2165.58	2126023	3600.000	12.12	0.000258
scpe3	7.00	4.78	123617.20	116806	3600.000	10.73	0.112
scpe4	110.00	4.39	259035.63	51796	3600.020	8.29	0.404
scpe5	7.00	4.75	46511.62	109950	3600.020	12.76	0.112
shmean	178.80	—	22547.43	3110	1346.422	250.73	1.24

Table 14: Detailed results for the USCCSP instances with random seed 3 using *LNS check*.

	Primal bound	Dual bound	Primal integral	Nodes	Time (sec)	Iter per node	BC per node
scp41	431.00	431.00	3701.81	1	60.580	829.00	23
scp410	516.00	516.00	3181.01	1	48.490	584.00	5
scp42	528.00	515.89	19394.33	8143	3600.010	245.69	0.341
scp43	525.00	525.00	14768.11	12042	1756.880	241.08	0.0913
scp44	496.00	496.00	3526.90	1	60.910	875.00	6
scp45	517.00	517.00	3294.23	945	108.190	8.34	0.0984
scp46	614.00	-99397.71	64302.88	495	3600.010	8442.06	1.97
scp47	436.00	436.00	50897.32	3589	1027.360	48.90	0.123
scp48	494.00	494.00	5457.43	133	103.940	21.02	0.173
scp49	897.00	-198891.94	102856.99	421	3600.110	5739.24	1.28
scp61	173.00	139.01	53031.77	3451	3600.060	720.19	1.14
scp62	161.00	147.00	71591.64	8226	3600.060	182.08	0.47
scp63	372.00	-99585.32	198230.47	2753	3600.000	207.73	0.564
scp64	164.00	132.02	97211.90	4636	3600.070	152.70	1.08
scp65	175.00	161.92	34467.42	9325	3600.000	738.39	0.267
scpclr10	27.00	21.33	25472.04	25778	3600.000	1327.21	0.18
scpclr11	39.00	17.92	82310.37	8333	3600.000	1868.80	0.19
scpclr12	41.00	17.12	179971.53	7471	3600.000	967.83	0.156
scpclr13	49.00	14.35	170895.34	539	3600.010	2351.13	0.497
scpcyc06	62.00	54.77	15402.98	1152980	3600.020	67.48	4.25e-05
scpcyc07	159.00	117.19	17807.74	59135	3600.010	486.24	5.07e-05
scpcyc08	355.00	257.81	7686.11	3042	3600.000	3165.44	0.000329
scpcyc09	945.00	576.50	48673.20	304	3600.030	5949.73	0.00329
scpcyc10	5120.00	—	315750.00	1	3606.190	0.00	0
scpe1	10.00	4.49	333416.29	53143	3600.140	9.10	0.371
scpe2	49.00	4.50	324569.11	21984	3600.000	30.10	0.935
scpe3	106.00	4.41	339932.56	56444	3600.030	8.30	0.376
scpe4	141.00	4.37	346119.50	43870	3600.160	9.13	0.527
scpe5	6.00	4.87	121402.37	125340	3600.000	12.44	0.0932
shmean	203.19	—	44575.79	2807	1750.697	262.79	1.13

Table 15: Detailed results for the USCCSP instances with random seed 5 using *LNS check*.

	Primal bound	Dual bound	Primal integral	Nodes	Time (sec)	Iter per node	BC per node
scp41	432.00	432.00	25186.08	4	739.030	16042.50	82.5
scp410	521.00	521.00	3577.89	856	91.220	90.70	0.0981
scp42	759.00	-199166.14	117727.30	466	3600.100	2565.93	1.53
scp43	527.00	527.00	5816.57	4800	363.520	68.45	0.0983
scp44	494.00	494.00	3495.37	1	56.730	1394.00	3
scp45	546.00	-99466.74	23166.88	485	3600.010	3708.76	1.61
scp46	563.00	563.00	3490.20	88	79.310	29.10	0.239
scp47	459.00	-99553.20	24465.65	775	3600.000	3433.38	1.5
scp48	497.00	497.00	22657.03	80	1155.150	1677.84	8.97
scp49	642.00	642.00	9229.60	18	254.670	1719.11	10.4
scp61	174.00	138.96	152534.15	6352	3600.000	219.87	0.415
scp62	183.00	146.62	111179.44	2708	3600.000	300.74	1.52
scp63	160.00	148.81	23960.83	23578	3600.000	123.98	0.119
scp64	150.00	131.00	101331.49	2478	3600.000	3937.93	0.626
scp65	211.00	161.00	79005.44	2700	3600.030	294.76	1.5
scpclr10	36.00	21.41	100774.33	12352	3600.010	2279.90	1.1
scpclr11	40.00	16.96	169058.75	2093	3600.010	6021.36	1.44
scpclr12	38.00	17.26	161447.85	7018	3600.000	1007.61	0.147
scpclr13	48.00	14.46	134619.21	548	3600.000	2419.47	0.398
scpcyc06	60.00	54.19	2475.10	1739123	3600.010	40.70	5.64e-05
scpcyc07	152.00	117.85	96.01	66532	3600.000	490.53	1.5e-05
scpcyc08	355.00	257.86	7665.06	3069	3600.000	3160.84	0.000326
scpcyc09	945.00	576.50	42481.81	313	3600.010	6113.36	0.00319
scpcyc10	5120.00	—	316230.00	1	3608.740	0.00	0
scpe1	47.00	4.50	326044.65	33394	3600.120	14.51	0.586
scpe2	106.00	4.38	339846.67	52206	3600.050	12.11	0.378
scpe3	6.00	4.92	70767.29	214596	3600.000	10.27	0.0376
scpe4	139.00	4.37	345423.09	46262	3600.040	10.05	0.556
scpe5	35.00	4.45	331721.84	34328	3600.010	18.32	0.695
shmean	205.61	—	37766.80	2166	1874.793	404.51	1.97

Table 16: Detailed results for the USCCSP instances with random seed 7 using *LNS check*.

	Primal bound	Dual bound	Primal integral	Nodes	Time (sec)	Iter per node	BC per node
scp41	429.00	429.00	1055.24	1	31.030	858.00	3
scp410	803.00	-598896.98	133893.92	406	3600.070	2979.26	1.26
scp42	517.00	517.00	2429.16	35	49.580	67.26	0.486
scp43	516.00	516.00	801.51	1	12.510	222.00	2
scp44	502.00	502.00	18626.69	209	744.200	511.04	2.36
scp45	515.00	515.00	1137.08	1	39.490	775.00	2
scp46	563.00	563.00	1497.24	62	63.800	29.87	0.581
scp47	713.00	-99319.43	140101.27	887	3600.000	4202.94	1.24
scp48	497.00	497.00	890.52	53	27.360	35.32	0.151
scp49	644.00	644.00	2212.71	137	84.690	90.34	0.803
scp61	144.00	142.01	10179.69	21339	3600.000	1015.86	0.034
scp62	153.00	153.00	6936.11	71567	1574.680	22.01	0.0107
scp63	175.00	150.11	37897.82	27713	3600.010	110.93	0.205
scp64	137.00	134.01	20354.02	67796	3600.000	95.42	0.0271
scp65	166.00	164.02	11049.51	55336	3600.050	83.02	0.0378
scpclr10	41.00	21.34	88318.12	12856	3600.010	2092.09	0.979
scpclr11	27.00	18.25	98429.57	26876	3600.000	544.13	0.069
scpclr12	42.00	17.14	74764.36	7678	3600.000	966.75	0.176
scpclr13	35.00	14.45	165531.48	436	3600.020	2524.72	0.495
scpcyc06	62.00	53.13	11661.11	420109	3600.010	265.54	0.000464
scpcyc07	152.00	117.86	77.77	66885	3600.020	490.08	1.5e-05
scpcyc08	377.00	258.00	26982.53	2800	3600.010	3191.12	0.000357
scpcyc09	873.00	576.43	42056.89	162	3600.030	3459.22	0.00617
scpcyc10	5120.00	—	315990.00	1	3613.580	0.00	0
scpe1	6.00	4.95	20632.28	125246	3600.010	19.68	0.149
scpe2	6.00	4.55	200066.91	123601	3600.000	12.56	0.0847
scpe3	5.00	5.00	44162.87	42403	980.990	18.06	0.111
scpe4	70.00	-49990.10	206075.96	1	3600.010	4651313.00	2.53e+03
scpe5	5.00	5.00	5489.42	18945	111.710	20.85	0.0319
shmean	167.99	—	15569.58	1747	993.777	341.40	2.75

Table 17: Detailed results for the USCCSP instances with random seed 1 using *Trust Region*.

	Primal bound	Dual bound	Primal integral	Nodes	Time (sec)	Iter per node	BC per node
scp41	432.00	432.00	2757.14	101	78.080	26.08	0.356
scp410	514.00	514.00	1026.37	1	14.180	167.00	2
scp42	514.00	514.00	1592.79	1	53.180	1355.00	14
scp43	518.00	518.00	24285.29	1	802.690	92646.00	436
scp44	494.00	494.00	695.03	1	15.050	219.00	1
scp45	519.00	519.00	2016.23	760	83.130	10.39	0.0526
scp46	563.00	563.00	2191.76	223	69.560	25.58	0.224
scp47	436.00	436.00	4364.09	1060	140.990	15.70	0.0425
scp48	501.00	501.00	2134.48	5578	106.310	18.43	0.0228
scp49	697.00	-2696053.86	29270.79	18952	3600.010	215.09	0.0374
scp61	832.00	-897987.25	298868.05	872	3600.010	1020.24	1.2
scp62	150.00	150.00	85959.81	15958	1637.450	87.13	0.055
scp63	152.00	152.00	6550.12	11692	145.570	13.87	0.00778
scp64	141.00	134.00	176854.94	38100	3600.000	75.28	0.0558
scp65	182.00	162.03	39397.63	7727	3600.010	493.83	0.499
scpclr10	27.00	21.65	4575.27	61151	3600.010	687.68	0.0477
scpclr11	30.00	18.29	46561.01	34253	3600.000	411.55	0.06
scpclr12	33.00	17.25	40622.28	8323	3600.000	910.80	0.181
scpclr13	52.00	14.42	164681.25	753	3600.350	2071.97	0.425
scpcyc06	61.00	52.93	11279.05	437906	3600.070	222.48	0.00129
scpcyc07	152.00	117.85	71.73	66626	3600.000	490.54	1.5e-05
scpcyc08	377.00	258.00	27105.20	2797	3600.000	3189.81	0.000358
scpcyc09	871.00	576.43	37382.27	162	3600.040	3459.22	0.00617
scpcyc10	5120.00	—	315780.00	1	3613.020	0.00	0
scpe1	6.00	4.85	168870.70	69509	3600.010	19.17	0.171
scpe2	35.00	4.55	301057.16	22560	3600.100	36.26	0.872
scpe3	40.00	4.59	100000.70	29151	3600.310	34.11	0.825
scpe4	7.00	4.55	290742.87	43930	3600.000	19.21	0.531
scpe5	6.00	4.32	268002.37	55865	3600.000	18.62	0.227
shmean	177.09	—	20039.51	2496	989.804	177.46	2.1

Table 18: Detailed results for the USCCSP instances with random seed 2 using *Trust Region*.

	Primal bound	Dual bound	Primal integral	Nodes	Time (sec)	Iter per node	BC per node
scp41	549.00	-199141.49	76544.57	504	3600.080	2239.01	1.59
scp410	519.00	519.00	1968.86	1	59.730	858.00	43
scp42	518.00	518.00	3529.24	3185	143.740	50.36	0.0797
scp43	519.00	-99418.96	2805.43	322	3600.000	17890.02	1.62
scp44	719.00	-199316.87	110566.04	470	3600.010	3452.00	1.44
scp45	521.00	521.00	2088.57	2182	78.010	19.78	0.0545
scp46	561.00	561.00	1270.63	1	46.360	259.00	8
scp47	647.00	-398452.08	119200.74	1144	3600.040	626.28	1.26
scp48	503.00	503.00	5753.35	5113	346.430	31.57	0.0475
scp49	646.00	646.00	10146.76	270	511.670	207.27	0.859
scp61	174.00	139.52	74670.01	6565	3600.030	234.98	0.671
scp62	199.00	149.71	143737.82	5315	3600.000	119.82	0.669
scp63	152.00	149.06	76500.07	86438	3600.410	87.90	0.0169
scp64	133.00	133.00	60627.52	708	902.870	142.64	0.629
scp65	175.00	162.01	47860.23	9686	3600.000	429.79	0.279
scpclr10	29.00	21.33	32086.04	13148	3600.000	1950.18	0.465
scpclr11	25.00	18.29	66005.15	36841	3600.010	406.85	0.0321
scpclr12	35.00	17.20	13804.72	8215	3600.000	938.44	0.103
scpclr13	56.00	14.45	133082.84	650	3600.000	2586.89	0.722
scpcyc06	62.00	54.79	11915.44	1075081	3600.020	69.44	5.77e-05
scpcyc07	152.00	117.86	72.23	66898	3600.000	490.04	1.49e-05
scpcyc08	377.00	258.00	27071.01	2803	3600.000	3190.58	0.000357
scpcyc09	873.00	576.43	35709.84	163	3600.000	3438.01	0.00613
scpcyc10	5120.00	—	315520.00	1	3614.080	0.00	0
scpe1	97.00	-399968.00	334070.12	1	3600.040	86985.00	1.51e+04
scpe2	6.00	5.18	30532.50	468459	3600.000	13.27	0.00756
scpe3	20.00	4.48	301542.30	30686	3600.020	24.33	0.755
scpe4	106.00	4.39	249827.61	54747	3600.020	9.06	0.367
scpe5	33.00	4.14	299999.36	59480	3600.000	11.25	0.248
shmean	196.79	—	29676.68	2604	1764.389	325.41	4.47

Table 19: Detailed results for the USCCSP instances with random seed 3 using *Trust Region*.

	Primal bound	Dual bound	Primal integral	Nodes	Time (sec)	Iter per node	BC per node
scp41	431.00	431.00	1969.84	60	92.330	23.07	0.967
scp410	516.00	516.00	2653.81	1	280.030	2109.00	120
scp42	530.00	518.02	16253.17	11639	3600.000	291.37	0.188
scp43	525.00	525.00	4794.10	5898	779.540	59.65	0.0778
scp44	632.00	-98797.13	78314.62	502	3600.000	5968.35	1.93
scp45	517.00	517.00	2333.50	35	114.660	46.20	0.857
scp46	564.00	564.00	24021.62	1	296.780	1707.00	83
scp47	436.00	436.00	30004.52	3502	916.250	198.72	0.157
scp48	494.00	494.00	4069.15	37	78.440	37.86	0.324
scp49	646.00	646.00	8750.13	2716	792.990	584.68	0.163
scp61	164.00	139.01	67879.27	20580	3600.000	254.30	0.132
scp62	163.00	148.26	31752.46	12883	3600.000	140.19	0.243
scp63	428.00	-99783.42	219744.90	1641	3600.040	442.15	0.873
scp64	176.00	132.01	112834.42	6758	3600.010	375.50	0.551
scp65	178.00	161.05	66556.49	4748	3600.040	801.38	0.691
scpclr10	30.00	21.33	64674.70	35953	3600.000	959.96	0.193
scpclr11	43.00	16.89	131604.26	2060	3600.000	6002.68	1.54
scpclr12	23.00	17.08	90591.76	8224	3600.010	963.07	0.0758
scpclr13	46.00	14.33	106996.10	853	3600.010	1738.74	0.143
scpcyc06	60.00	54.09	635.26	2498504	3600.000	35.70	4e-06
scpcyc07	152.00	117.46	146.12	64500	3600.000	470.59	6.2e-05
scpcyc08	377.00	258.00	27221.39	2799	3600.010	3192.21	0.000357
scpcyc09	873.00	576.43	31764.49	162	3600.080	3459.22	0.00617
scpcyc10	5120.00	—	314650.00	1	3608.130	0.00	0
scpe1	56.00	4.19	318614.21	57951	3600.080	14.92	0.264
scpe2	5.00	5.00	47385.50	185762	2035.910	11.86	0.0212
scpe3	132.00	-99957.69	343672.22	1	3600.000	2783875.00	4.21e+03
scpe4	6.00	4.87	96644.45	135437	3600.020	13.52	0.0692
scpe5	6.00	5.15	38418.45	446920	3600.000	12.17	0.00844
shmean	184.74	—	27581.20	2471	1765.811	367.59	4.99

Table 20: Detailed results for the USCCSP instances with random seed 5 using *Trust Region*.

	Primal bound	Dual bound	Primal integral	Nodes	Time (sec)	Iter per node	BC per node
scp41	432.00	432.00	2015.32	1	38.340	786.00	12
scp410	521.00	521.00	975.73	1142	54.620	11.88	0.0569
scp42	515.00	515.00	4360.57	1	137.160	2470.00	25
scp43	527.00	527.00	2223.61	10986	199.240	49.20	0.0183
scp44	494.00	494.00	1149.84	1	22.090	218.00	2
scp45	527.00	527.00	71362.90	3176	2055.880	1688.95	0.19
scp46	563.00	563.00	1917.85	391	84.700	16.52	0.0614
scp47	435.00	435.00	4715.22	1125	134.080	24.50	0.0978
scp48	497.00	497.00	6123.40	21	305.400	787.67	4.81
scp49	642.00	642.00	1776.82	1	477.950	40941.00	312
scp61	268.00	-99819.26	167917.46	661	3600.600	1242.22	1.74
scp62	163.00	146.02	58322.01	4704	3600.060	250.06	0.896
scp63	164.00	148.03	30440.49	18334	3600.020	230.87	0.173
scp64	348.00	129.00	278966.05	11	3600.000	488700.36	71.2
scp65	184.00	162.00	84886.15	3555	3600.010	449.55	0.821
scpclr10	28.00	21.44	41575.86	20870	3600.000	1509.14	0.587
scpclr11	23.00	18.34	66635.52	34696	3600.010	412.84	0.0346
scpclr12	39.00	17.07	163148.75	5009	3600.010	1392.56	0.172
scpclr13	41.00	14.45	115178.87	583	3600.050	2258.57	0.201
scpcyc06	61.00	54.62	6246.89	1123102	3600.020	68.88	2.49e-05
scpcyc07	152.00	117.86	76.25	67278	3600.010	489.13	1.49e-05
scpcyc08	377.00	258.00	27050.92	2797	3600.000	3189.21	0.000358
scpcyc09	871.00	576.43	36326.47	162	3600.020	3459.22	0.00617
scpcyc10	5120.00	—	316390.00	1	3612.420	0.00	0
scpe1	36.00	4.65	307871.91	23124	3600.060	34.56	0.868
scpe2	44.00	4.45	311529.55	16120	3600.000	32.44	1.09
scpe3	6.00	4.89	45705.56	201266	3600.000	10.56	0.0398
scpe4	50.00	4.50	318146.18	30293	3600.040	33.78	0.807
scpe5	43.00	4.51	310929.27	20363	3600.010	30.86	0.955
shmean	191.83	—	24102.81	1621	1231.161	360.96	3.65

Table 21: Detailed results for the USCCSP instances with random seed 7 using *Trust Region*.

	Primal bound	Dual bound	Primal integral	Nodes	Time (sec)	Iter per node	BC per node
scp41	439.00	-199497.57	12585.63	448	3600.000	4472.06	1.47
scp410	516.00	516.00	4488.42	1	69.580	685.00	11
scp42	517.00	517.00	4087.48	1	62.960	1149.00	29
scp43	537.00	-99395.69	16580.57	512	3600.130	5040.14	1.92
scp44	607.00	-99450.15	69811.47	2134	3600.010	987.32	0.554
scp45	515.00	515.00	2709.84	1	42.230	691.00	5
scp46	563.00	563.00	3977.61	146	73.840	36.15	0.452
scp47	438.00	438.00	31869.27	15276	1985.080	346.55	0.0354
scp48	609.00	-99396.98	68539.62	357	3600.010	12019.27	1.83
scp49	644.00	644.00	4631.24	30	87.900	47.67	0.333
scp61	144.00	144.00	7793.83	181855	3316.250	21.04	0.00606
scp62	154.00	149.95	37821.14	52291	3600.000	67.46	0.0581
scp63	161.00	149.03	28748.19	9870	3600.010	226.47	0.532
scp64	140.00	132.00	24393.85	2195	3600.010	1719.89	0.736
scp65	166.00	164.02	10121.30	71457	3600.000	73.68	0.0233
scpclr10	31.00	21.67	40719.85	19279	3600.020	1276.58	0.688
scpclr11	33.00	18.19	103146.29	24344	3600.000	607.59	0.0907
scpclr12	50.00	17.04	112765.97	3333	3600.020	2080.86	0.411
scpclr13	44.00	14.38	162524.69	616	3600.100	2143.18	0.659
scpcyc06	60.00	54.37	1139.01	2143847	3600.040	40.38	1.87e-05
scpcyc07	152.00	117.86	94.55	67384	3600.000	488.97	1.48e-05
scpcyc08	377.00	258.00	27467.15	2773	3600.010	3181.76	0.000361
scpcyc09	879.00	576.43	41285.46	162	3600.020	3459.22	0.00617
scpcyc10	5120.00	—	316290.00	1	3610.480	0.00	0
scpe1	6.00	4.66	299428.73	30003	3600.020	29.54	0.769
scpe2	5.00	5.00	4517.16	18076	100.550	22.28	0.0305
scpe3	37.00	4.62	311464.65	30361	3600.220	37.93	0.796
scpe4	30.00	4.46	53044.08	21548	3600.040	31.05	0.913
scpe5	5.00	5.00	18202.07	22176	311.000	19.93	0.0783
shmean	168.62	—	20849.91	2650	1467.876	317.79	1.35

Table 22: Detailed results for the USCCSP instances with random seed 1 using *TR LNS check*.

	Primal bound	Dual bound	Primal integral	Nodes	Time (sec)	Iter per node	BC per node
scp41	432.00	432.00	13722.08	478	372.540	95.13	0.341
scp410	514.00	514.00	4214.64	1	83.680	902.00	7
scp42	514.00	514.00	3690.83	1	80.510	1746.00	27
scp43	518.00	518.00	2854.40	1	43.160	797.00	11
scp44	494.00	494.00	5245.39	2	297.740	6331.50	51
scp45	663.00	-299136.20	80407.43	506	3600.000	3295.36	1.78
scp46	563.00	563.00	2702.82	2	55.440	430.00	10.5
scp47	456.00	-99547.21	21051.57	366	3600.000	9137.34	1.22
scp48	501.00	501.00	4217.99	1349	99.150	12.22	0.0363
scp49	641.00	641.00	2792.73	11	70.250	101.18	0.545
scp61	142.00	142.00	11789.06	3512	366.380	47.28	0.0803
scp62	150.00	150.00	12336.66	3257	685.520	104.62	0.27
scp63	152.00	149.01	12653.13	13861	3600.000	347.70	0.0765
scp64	141.00	133.01	11662.25	16303	3600.010	409.43	0.101
scp65	185.00	160.10	87167.53	2468	3600.000	2007.21	1.21
scpclr10	30.00	21.55	95064.65	15861	3600.000	1801.14	0.959
scpclr11	43.00	17.78	140533.80	7131	3600.010	1879.86	0.424
scpclr12	35.00	16.50	29846.89	1547	3600.010	3210.36	1.42
scpclr13	37.00	14.35	95012.30	505	3600.010	2341.41	0.259
scpcyc06	61.00	54.51	6158.87	689981	3600.000	118.62	2.61e-05
scpcyc07	152.00	117.86	81.33	67065	3600.010	489.56	1.49e-05
scpcyc08	377.00	258.00	27324.85	2743	3600.010	3182.64	0.000365
scpcyc09	881.00	576.43	44069.29	162	3600.030	3459.22	0.00617
scpcyc10	5120.00	—	315060.00	1	3612.740	0.00	0
scpe1	37.00	4.70	302675.55	27216	3600.160	38.48	0.818
scpe2	30.00	4.57	291278.88	23907	3600.090	36.51	0.877
scpe3	30.00	4.35	261513.03	49784	3600.050	11.57	0.485
scpe4	7.00	4.63	294686.98	29258	3600.100	37.38	0.81
scpe5	34.00	4.51	327401.63	26331	3663.420	32.84	0.829
shmean	179.65	—	24196.48	1286	1209.844	375.99	2.43

Table 23: Detailed results for the USCCSP instances with random seed 2 using *TR LNS check*.

	Primal bound	Dual bound	Primal integral	Nodes	Time (sec)	Iter per node	BC per node
scp41	549.00	-199443.49	78300.87	505	3600.140	3829.65	1.49
scp410	519.00	519.00	4204.03	1	69.920	708.00	25
scp42	518.00	518.00	17574.55	290	404.170	187.30	1.03
scp43	517.00	517.00	2975.91	1	65.610	1799.00	13
scp44	500.00	500.00	4640.87	155	92.150	21.23	0.458
scp45	521.00	521.00	15539.55	3970	595.560	33.64	0.0972
scp46	561.00	561.00	3066.33	1	45.030	727.00	7
scp47	647.00	-99169.93	120397.50	604	3600.000	8987.51	1.24
scp48	503.00	503.00	3638.12	22811	185.250	13.88	0.00583
scp49	646.00	646.00	3780.77	126	91.860	19.52	0.111
scp61	179.00	138.87	30807.91	2683	3600.000	627.62	1.41
scp62	176.00	150.51	41332.83	12807	3600.000	668.70	0.173
scp63	159.00	147.21	51712.33	6979	3600.970	334.04	0.515
scp64	133.00	133.00	6084.01	444	99.730	46.39	0.241
scp65	174.00	162.01	27820.83	5863	3600.060	684.52	0.567
scpclr10	27.00	21.38	12927.79	39091	3600.010	1106.47	0.104
scpclr11	23.00	18.29	30006.42	34585	3600.010	419.58	0.0105
scpclr12	40.00	17.27	53534.21	8180	3600.010	944.16	0.112
scpclr13	41.00	14.48	90703.22	597	3600.000	2127.61	0.496
scpcyc06	61.00	54.82	8406.75	1068371	3600.010	70.91	4.49e-05
scpcyc07	152.00	117.86	89.46	67200	3600.000	489.25	1.49e-05
scpcyc08	377.00	258.00	27419.04	2775	3600.000	3183.58	0.00036
scpcyc09	881.00	576.43	42797.95	162	3600.030	3459.22	0.00617
scpcyc10	5120.00	—	314870.00	1	3608.670	0.00	0
scpe1	46.00	4.64	305501.02	33680	3600.020	32.30	0.723
scpe2	6.00	4.98	94093.46	189930	3600.010	13.72	0.0374
scpe3	16.00	4.60	313819.05	29494	3600.030	34.30	0.815
scpe4	33.00	4.59	20390.32	28763	3601.300	38.90	0.803
scpe5	32.00	4.51	282582.46	18228	3600.090	35.57	0.969
shmean	179.78	—	23311.21	2314	1302.079	253.45	1.36

Table 24: Detailed results for the USCCSP instances with random seed 3 using *TR LNS check*.

	Primal bound	Dual bound	Primal integral	Nodes	Time (sec)	Iter per node	BC per node
scp41	431.00	431.00	3701.81	1	60.590	829.00	23
scp410	516.00	516.00	3176.38	1	48.370	584.00	5
scp42	531.00	516.08	14995.93	14861	3600.010	146.06	0.161
scp43	525.00	525.00	6529.99	4824	546.560	82.00	0.0738
scp44	496.00	496.00	3526.90	1	60.890	875.00	6
scp45	517.00	517.00	2950.01	47	68.070	28.04	0.574
scp46	643.00	-99397.77	69207.41	633	3600.170	1894.73	1.79
scp47	436.00	436.00	53297.76	2768	1132.520	79.30	0.172
scp48	494.00	494.00	5345.40	38	98.900	40.50	0.289
scp49	897.00	-198888.99	102846.84	589	3600.000	3275.92	1.17
scp61	205.00	138.00	86164.32	2293	3600.010	179.74	1.57
scp62	156.00	149.02	19221.39	32555	3600.000	121.56	0.0849
scp63	372.00	-99585.03	198230.06	2956	3600.000	156.90	0.618
scp64	141.00	134.02	17652.52	25437	3600.000	257.85	0.0664
scp65	175.00	162.80	24625.30	12773	3600.010	520.59	0.173
scpclr10	31.00	21.40	57240.65	25820	3600.010	1345.99	0.4
scpclr11	37.00	16.93	65350.50	1731	3600.000	6119.81	1.37
scpclr12	25.00	17.26	105653.74	7775	3600.010	976.90	0.095
scpclr13	38.00	14.47	151883.16	486	3600.000	2607.59	0.737
scpcyc06	62.00	55.00	11738.83	1144850	3600.040	70.86	2.88e-05
scpcyc07	154.00	117.45	4808.08	67841	3600.000	443.66	8.84e-05
scpcyc08	377.00	258.00	27510.93	2767	3600.010	3178.73	0.000361
scpcyc09	884.00	576.43	44218.74	163	3600.000	3439.53	0.00613
scpcyc10	5120.00	—	314520.00	1	3610.160	0.00	0
scpe1	106.00	4.50	333480.49	55352	3600.040	9.31	0.364
scpe2	130.00	4.40	346342.82	54118	3600.020	8.06	0.399
scpe3	6.00	5.27	8728.96	955521	3600.000	11.08	0.00196
scpe4	108.00	4.38	340253.70	55589	3600.100	8.75	0.373
scpe5	17.00	4.59	237927.84	26633	3600.170	38.73	0.808
shmean	206.14	—	34589.21	2750	1660.806	236.64	1.15

Table 25: Detailed results for the USCCSP instances with random seed 5 using *TR LNS check*.

	Primal bound	Dual bound	Primal integral	Nodes	Time (sec)	Iter per node	BC per node
scp41	432.00	432.00	25186.08	4	738.800	16042.50	82.5
scp410	521.00	521.00	3634.16	2220	114.620	16.93	0.0441
scp42	752.00	-199135.72	116119.57	443	3600.000	5926.08	1.46
scp43	527.00	527.00	6501.91	2012	305.800	68.52	0.168
scp44	494.00	494.00	3484.94	1	56.610	1394.00	3
scp45	545.00	-99467.07	17402.08	732	3600.010	2888.35	1.58
scp46	563.00	563.00	3441.27	51	67.390	32.63	0.235
scp47	460.00	-99552.28	25590.06	483	3600.010	5266.82	1.52
scp48	497.00	497.00	22756.80	67	1183.860	1952.52	10.7
scp49	642.00	642.00	9234.93	19	256.280	1614.37	10.2
scp61	148.00	142.01	94883.40	26099	3600.000	210.85	0.0816
scp62	213.00	145.40	97101.41	2719	3600.010	123.79	1.47
scp63	187.00	147.20	61561.08	4795	3600.050	682.47	0.72
scp64	151.00	131.01	97228.09	1789	3600.010	4619.95	0.746
scp65	194.00	161.75	45854.79	4630	3600.070	404.56	0.844
scpclr10	31.00	21.53	52656.05	12191	3600.010	2496.67	0.589
scpclr11	39.00	17.82	152956.72	7457	3600.010	1803.49	0.338
scpclr12	39.00	17.28	159650.56	8036	3600.020	939.69	0.144
scpclr13	45.00	14.38	119196.51	868	3600.010	1672.57	0.122
scpcyc06	62.00	54.95	11779.36	1126834	3600.010	71.80	1.95e-05
scpcyc07	152.00	117.86	93.99	66808	3600.020	490.21	1.5e-05
scpcyc08	377.00	258.00	27484.37	2778	3600.000	3184.35	0.00036
scpcyc09	879.00	576.43	37103.66	162	3600.040	3459.22	0.00617
scpcyc10	5120.00	—	316290.00	1	3609.390	0.00	0
scpe1	6.00	4.87	168696.02	72439	3600.020	18.38	0.19
scpe2	6.00	4.80	120351.05	106416	3600.010	14.12	0.0922
scpe3	48.00	4.57	317646.58	31532	3600.140	33.08	0.8
scpe4	6.00	4.70	192176.49	62792	3600.000	14.72	0.198
scpe5	6.00	4.84	140732.08	87758	3600.000	16.99	0.151
shmean	169.44	—	37426.07	2235	1869.947	413.37	1.89

Table 26: Detailed results for the USCCSP instances with random seed 7 using *TR LNS check*.

	Primal bound	Dual bound	Primal integral	Nodes	Time (sec)	Iter per node	BC per node
cap101	890125.75	768267.73	854.18	22052	3627.930	70.47	48.9
cap102	952862.40	773090.38	613.47	21802	3674.350	86.87	43.8
cap103	1004163.46	788719.78	4169.75	24663	3600.050	86.81	32.4
cap104	1034438.83	824288.69	1996.84	25510	3600.050	86.28	29
cap111	950752.73	836694.30	2912.23	6500	3600.000	160.10	18.1
cap112	1037461.58	902271.90	4320.34	7325	3600.070	168.45	12
cap113	1103935.81	967809.73	7418.95	8066	3600.010	184.00	9.26
cap114	1202836.05	1065538.64	3878.70	8962	3600.080	221.82	7.25
cap121	911292.51	740596.90	7918.48	9276	3600.140	113.15	14.3
cap122	1000808.23	766393.91	17352.16	9803	3600.010	126.02	11.5
cap123	1037163.12	791948.63	13180.93	11073	3600.000	104.66	11
cap124	1059156.15	829017.77	16202.36	11070	3600.030	128.15	9.21
cap131	942532.40	716289.08	13299.67	24447	3600.040	64.93	15.5
cap132	969883.43	732851.37	1814.62	23850	3600.050	68.82	16.2
cap133	1021627.71	742836.33	10627.66	21564	3600.050	78.11	15.1
cap134	1053124.92	757822.41	12380.50	23131	3600.080	75.95	12.7
cap81	939586.37	891727.18	1109.59	5092	3600.000	287.22	38.9
cap82	1018711.90	958669.13	704.70	4703	3600.000	290.98	36.3
cap83	1093625.32	1031107.12	755.41	5120	3600.010	292.19	30.6
cap84	1200032.03	1131169.77	1871.33	5121	3600.010	277.78	29.6
cap91	893342.96	776476.78	4576.25	9363	3600.010	140.83	27.8
cap92	964053.14	802461.52	4649.19	7251	3607.710	177.22	39.6
cap93	1002459.28	831376.32	4204.06	8123	3613.900	181.96	31.9
cap94	1061532.56	870591.04	7681.18	7954	3600.100	192.39	28.8
shmean	1011283.86	839199.54	3915.29	10975	3605.161	136.81	21.6

Table 27: Detailed results for the SCFLP instances with random seed 1 using *Benders*.

	Primal bound	Dual bound	Primal integral	Nodes	Time (sec)	Iter per node	BC per node
cap101	901685.92	771316.39	3417.72	23505	3600.060	72.95	46.5
cap102	973933.76	781548.67	3120.51	22292	3601.640	80.46	44.7
cap103	1009757.16	796277.27	4649.44	24297	3600.070	89.54	34.2
cap104	1042764.53	829676.98	2617.95	27318	3600.000	85.20	26.4
cap111	965380.90	844762.66	5841.60	6396	3600.010	156.22	20.6
cap112	1048627.88	911022.87	9239.48	7275	3600.020	177.59	12.6
cap113	1112184.43	977064.02	7398.44	8052	3600.020	186.16	8.96
cap114	1219669.54	1074804.00	3444.04	6974	3600.000	243.99	10.5
cap121	897391.76	748666.97	2092.62	11893	3600.000	99.49	8.92
cap122	986041.75	774583.15	8485.58	9901	3600.010	120.08	11.4
cap123	1040438.15	799223.87	6502.35	10577	3600.100	119.77	11.3
cap124	1070898.68	838182.09	3294.04	11803	3600.010	119.39	8.24
cap131	925828.60	726677.08	10946.66	24498	3600.050	67.21	15.4
cap132	980234.93	738975.93	7827.49	21367	3600.000	75.32	14.9
cap133	1022177.66	747097.21	8331.16	20046	3600.000	79.09	15.3
cap134	1063209.13	769456.71	5912.59	28119	3600.000	71.35	11.8
cap81	945336.05	899827.64	1012.25	4388	3600.020	301.06	47
cap82	1032074.13	970516.41	2898.48	4372	3600.020	329.67	38.4
cap83	1100893.24	1036781.64	1745.66	4844	3600.090	294.58	32.3
cap84	1210569.34	1131936.88	2183.48	4840	3600.010	296.58	31.4
cap91	901422.45	784275.49	2459.71	9491	3600.010	140.91	29.4
cap92	965770.16	811876.66	2688.43	8770	3606.680	172.23	29.3
cap93	1002915.10	837003.05	2813.95	8077	3600.000	170.54	30
cap94	1057007.30	872881.18	2765.18	7250	3604.850	192.70	30
shmean	1016517.00	846649.55	3930.75	10942	3600.569	138.16	21.4

Table 28: Detailed results for the SCFLP instances with random seed 2 using *Benders*.

	Primal bound	Dual bound	Primal integral	Nodes	Time (sec)	Iter per node	BC per node
cap101	905822.96	771539.45	5101.46	22801	3600.080	76.81	58.3
cap102	964559.22	780491.02	315.09	23774	3623.120	84.85	40.4
cap103	1013696.11	793146.47	4215.76	24599	3600.070	86.89	33.5
cap104	1039008.01	815792.85	1743.87	24529	3600.000	91.15	28.5
cap111	960783.23	843655.82	3898.67	6395	3600.000	177.30	18.3
cap112	1051262.71	910831.67	9146.43	7827	3600.100	184.19	10.9
cap113	1121524.79	977743.30	9310.67	8167	3600.220	217.75	9.04
cap114	1229904.69	1074419.53	8324.94	9343	3600.020	213.54	8.23
cap121	925814.71	747115.85	14766.79	9864	3600.010	117.88	14.2
cap122	962437.12	771426.37	2487.38	9829	3600.220	119.67	13.1
cap123	1022576.61	799693.38	4241.21	12243	3600.010	101.56	9.44
cap124	1075396.90	833817.94	7297.38	10693	3600.000	122.63	9.56
cap131	935941.24	723788.15	807.41	23024	3600.080	64.72	17.2
cap132	982368.12	735253.35	1919.81	21821	3600.000	71.45	15.8
cap133	1015218.40	747974.17	9991.52	23792	3600.020	74.15	14.6
cap134	1051648.36	764614.58	16139.52	21274	3600.000	81.50	12.7
cap81	946036.00	901826.35	307.64	5407	3600.130	275.22	34.7
cap82	1031418.76	970840.15	2532.62	5072	3600.020	307.81	33.7
cap83	1101058.39	1042441.81	1088.41	5868	3600.020	284.25	26.3
cap84	1208049.14	1144580.04	580.41	7503	3600.010	269.21	16.9
cap91	894085.48	789294.38	1954.52	8822	3600.010	161.19	29.8
cap92	965846.48	810053.86	3243.03	7013	3608.250	174.42	33.7
cap93	1009101.96	834519.61	3806.40	7893	3600.010	190.23	31.1
cap94	1057971.04	882606.07	4651.73	9309	3600.000	178.62	21.5
shmean	1016342.67	846061.16	3159.44	11379	3601.347	139.39	20.5

Table 29: Detailed results for the SCFLP instances with random seed 3 using *Benders*.

	Primal bound	Dual bound	Primal integral	Nodes	Time (sec)	Iter per node	BC per node
cap101	897400.49	760564.65	1787.39	21686	3600.110	80.93	53
cap102	974654.42	780355.28	9066.62	23337	3600.080	87.25	46.7
cap103	1005163.83	792684.66	2653.90	24947	3600.060	86.16	35.4
cap104	1036713.77	813862.36	6151.94	22194	3600.060	96.94	30.8
cap111	949919.31	835064.75	1918.27	5897	3600.020	174.93	22.6
cap112	1036726.50	900533.53	6759.82	7435	3600.020	178.40	12.6
cap113	1106298.57	967096.21	7224.06	8383	3600.000	200.12	9
cap114	1205264.29	1065361.17	1324.44	9176	3600.080	226.01	7.29
cap121	909049.71	743091.18	4788.59	11382	3600.000	117.19	9.94
cap122	969116.76	770195.82	4102.14	12477	3600.090	115.38	8.87
cap123	1035200.63	797144.29	3530.27	14110	3600.170	106.22	7.93
cap124	1072301.88	829413.33	10441.20	9333	3600.120	151.95	11.2
cap131	933598.57	718206.28	7239.93	24093	3600.070	66.45	14.6
cap132	1008621.72	730581.38	4779.75	22561	3600.160	68.91	15.5
cap133	1023859.06	740374.58	11815.63	20551	3601.580	78.19	16.7
cap134	1058870.72	755935.74	9602.02	21091	3600.000	85.94	12.5
cap81	935995.74	891523.83	1191.09	4879	3600.230	303.39	38.1
cap82	1021099.88	957306.99	2856.12	4673	3600.000	330.50	34.9
cap83	1097042.74	1025792.39	3421.79	4610	3600.000	289.07	39.4
cap84	1196303.29	1127089.54	2887.05	5130	3600.160	305.39	27.6
cap91	895172.09	779005.53	5232.49	9321	3619.320	157.39	27.8
cap92	960364.81	805959.91	4856.76	7955	3811.470	173.89	35
cap93	1009887.36	835503.02	6870.42	7126	3721.770	206.95	31.4
cap94	1063298.96	873346.36	5995.79	9298	3600.000	184.85	23.3
shmean	1013618.97	839239.61	4453.54	11054	3614.510	143.97	21.4

Table 30: Detailed results for the SCFLP instances with random seed 5 using *Benders*.

	Primal bound	Dual bound	Primal integral	Nodes	Time (sec)	Iter per node	BC per node
cap101	904749.07	765921.81	5804.19	22239	3600.340	61.42	42.4
cap102	973652.80	779841.42	5767.83	23792	3600.130	85.95	46.1
cap103	1004956.85	789560.21	3002.81	26167	3600.080	89.20	32.5
cap104	1033751.84	816688.81	2244.15	26876	3603.300	85.79	24.7
cap111	949653.11	836816.08	5337.60	6714	3600.170	177.31	17.1
cap112	1038180.31	901928.93	5085.53	6330	3600.230	215.70	13.6
cap113	1104817.56	966803.41	5193.65	9860	3600.180	178.99	6.88
cap114	1205526.65	1065989.91	3498.13	6665	3600.000	247.29	11.1
cap121	915557.70	741595.00	7265.33	10785	3601.210	115.16	10.8
cap122	983403.63	767230.33	8228.36	11701	3600.000	121.25	9.47
cap123	1034150.99	791817.71	9932.19	9798	3600.220	131.67	11.7
cap124	1085768.99	829597.47	14540.47	11268	3600.010	122.50	9.25
cap131	941125.83	718839.51	13458.54	22779	3600.030	67.73	17
cap132	1001515.51	732620.23	5881.11	25810	3600.070	67.96	14.6
cap133	1068914.06	742866.87	17556.73	20505	3600.070	81.07	16.6
cap134	1047595.55	761330.34	12329.89	20902	3600.000	80.19	13.2
cap81	940856.17	889957.04	950.91	4654	3600.130	289.61	41
cap82	1025380.03	959624.94	2966.48	4094	3600.010	310.26	46.9
cap83	1097429.98	1025402.90	3099.20	4403	3600.010	297.73	39.7
cap84	1202943.04	1127971.72	1439.22	4906	3600.010	295.18	32.1
cap91	890872.06	779301.56	513.48	9410	3600.000	164.34	23.8
cap92	957912.37	805322.99	2059.56	8721	3627.140	176.31	27.2
cap93	1004277.67	835831.66	4701.67	9094	3600.050	177.45	24.2
cap94	1060572.03	880322.07	2966.82	10553	3600.110	194.35	19.2
shmean	1016594.61	839960.01	4465.31	11086	3601.392	141.97	20.8

Table 31: Detailed results for the SCFLP instances with random seed 7 using *Benders*.

	Primal bound	Dual bound	Primal integral	Nodes	Time (sec)	Iter per node	BC per node
cap101	898511.02	768139.31	3655.78	22880	3600.110	70.48	51.2
cap102	953512.35	770382.05	813.74	22435	3600.000	82.44	43.2
cap103	1001071.23	785285.41	2389.84	24278	3600.000	86.84	33.2
cap104	1035245.94	812901.66	3620.79	24636	3612.450	86.89	29.7
cap111	944425.28	836607.83	571.90	6870	3600.000	160.61	16.4
cap112	1026531.28	902165.87	611.83	7134	3600.000	156.76	13.3
cap113	1105360.99	967402.57	6645.92	7220	3600.010	188.46	11.3
cap114	1207600.13	1065218.44	5275.78	9880	3600.000	204.77	6.8
cap121	897453.53	739644.55	2668.27	10181	3600.010	89.75	12.4
cap122	987133.82	766137.30	12826.23	10287	3600.230	109.82	11.9
cap123	1004285.89	790394.96	1299.28	11408	3600.170	94.62	9.83
cap124	1103962.28	826189.26	19889.14	10643	3600.110	118.31	10
cap131	909358.14	715364.81	3274.05	20120	3603.330	53.68	16.4
cap132	1014727.45	726869.74	16501.78	20161	3602.540	65.88	15.4
cap133	1019860.40	738580.59	3231.21	20626	3600.100	70.29	15.6
cap134	1056881.56	753544.32	10039.44	20762	3602.600	74.95	14
cap81	943923.91	888475.40	2195.69	3960	3602.800	282.20	49
cap82	1018211.26	957914.55	662.75	5361	3600.010	274.50	32.4
cap83	1098147.19	1030745.68	2145.83	4761	3600.000	282.86	36.6
cap84	1196119.38	1131911.58	1336.04	5291	3600.000	281.05	27.5
cap91	896299.04	776895.37	4932.58	10145	3600.810	143.18	27.6
cap92	956515.70	800594.95	5991.65	6619	3600.020	184.06	43.1
cap93	999108.40	830079.41	1964.31	8472	3600.020	168.24	28.9
cap94	1052975.05	870630.52	4390.88	8632	3600.060	183.43	25
shmean	1010276.15	837165.86	3131.06	10793	3601.057	130.07	21.8

Table 32: Detailed results for the SCFLP instances with random seed 1 using *LNS check*.

	Primal bound	Dual bound	Primal integral	Nodes	Time (sec)	Iter per node	BC per node
cap101	908040.73	770813.26	4670.46	22778	3600.040	73.93	46.9
cap102	988198.63	778407.56	6398.11	23524	3615.440	83.23	43.5
cap103	1007867.32	795526.87	2386.49	22543	3600.080	89.63	37.7
cap104	1042764.53	822662.27	2618.19	24927	3600.000	84.53	28.8
cap111	962217.56	844650.87	4720.16	6469	3600.130	143.67	21.6
cap112	1041916.36	910554.40	7035.39	7340	3600.180	170.62	12.7
cap113	1097938.59	977679.95	2938.89	9574	3600.000	172.36	6.81
cap114	1220230.45	1073984.87	3189.09	9719	3600.010	206.37	6.81
cap121	901771.44	746823.29	2660.53	11174	3600.220	82.66	10.3
cap122	991860.64	772672.49	9704.86	9711	3600.000	98.40	12.3
cap123	1035742.89	797086.82	4838.68	9289	3600.100	105.57	12.4
cap124	1084730.81	836103.90	7901.77	11095	3600.000	109.41	9.19
cap131	928127.66	723728.62	9079.48	22540	3601.900	54.57	15.4
cap132	994359.08	733167.95	11490.12	17590	3600.040	70.94	16.4
cap133	1068933.37	745665.54	16280.32	20334	3600.010	75.80	16.2
cap134	1069392.72	762335.75	6081.01	23226	3600.000	74.20	13.1
cap81	945336.05	900105.95	851.70	4426	3600.010	300.42	43.6
cap82	1032183.76	967975.52	2787.31	4339	3600.010	313.55	43.3
cap83	1100676.99	1036871.63	1571.03	4773	3600.010	290.79	32
cap84	1210981.47	1131732.39	2256.53	4935	3600.010	269.49	32
cap91	901422.45	784309.72	2459.86	9467	3600.040	140.93	29.1
cap92	975062.60	808167.45	5108.57	8091	3671.380	167.04	31
cap93	1014655.32	837987.45	8122.76	7214	3600.030	187.56	35.3
cap94	1057007.30	875235.53	2984.39	7990	3617.640	178.94	27.4
shmean	1020990.76	844844.15	4316.71	10743	3604.441	130.59	21.9

Table 33: Detailed results for the SCFLP instances with random seed 2 using *LNS check*.

	Primal bound	Dual bound	Primal integral	Nodes	Time (sec)	Iter per node	BC per node
cap101	905822.96	771539.45	5100.88	23721	3600.040	75.88	56.7
cap102	978670.80	775472.71	5405.14	24494	3600.050	89.23	42
cap103	1007075.59	791738.60	2317.32	23602	3614.430	83.15	33.9
cap104	1039008.01	817264.35	3121.32	25504	3600.000	87.14	28.5
cap111	953186.99	843625.03	1248.73	6538	3600.050	173.14	17.4
cap112	1035598.41	911178.87	3710.19	9436	3600.010	178.34	7.83
cap113	1108615.45	977969.51	5409.12	9118	3600.010	204.15	7.2
cap114	1203632.54	1074053.04	850.19	11131	3600.000	214.80	5.29
cap121	908006.76	744834.17	11288.89	9596	3600.040	97.59	13.2
cap122	964425.09	770607.18	1831.29	9955	3600.010	110.00	11.9
cap123	1019013.76	797799.78	2546.62	12120	3600.000	98.09	9.31
cap124	1076176.79	832202.69	8546.93	9322	3600.090	122.12	11.9
cap131	945728.95	722163.67	4512.73	23509	3600.060	54.22	14.3
cap132	1014501.30	732353.83	11989.06	21687	3600.010	66.72	17
cap133	1034613.49	742392.50	13755.01	19772	3601.970	72.13	17.8
cap134	1061349.52	759159.26	9196.69	21698	3600.070	74.14	12.7
cap81	947280.62	899796.66	1409.65	4955	3600.080	290.56	36.9
cap82	1029858.71	969484.72	2850.56	4925	3600.010	308.06	34.3
cap83	1105936.47	1042002.66	1924.98	5717	3600.010	289.99	29.2
cap84	1208049.14	1147057.26	258.88	8285	3600.000	264.63	15.4
cap91	902637.35	789036.41	4662.32	9037	3600.020	159.12	31.5
cap92	968796.67	809364.19	2872.24	7987	3615.100	183.15	32.5
cap93	1007337.87	834666.99	4897.11	8381	3600.030	174.45	29.1
cap94	1052288.30	881770.87	2978.88	10750	3600.000	168.22	18
shmean	1016854.31	844675.87	3442.80	11649	3601.335	134.44	19.9

Table 34: Detailed results for the SCFLP instances with random seed 3 using *LNS check*.

	Primal bound	Dual bound	Primal integral	Nodes	Time (sec)	Iter per node	BC per node
cap101	899002.07	760355.28	2217.37	22776	3600.060	77.55	53.6
cap102	956958.89	780355.28	502.98	24017	3600.120	83.50	44.4
cap103	1008692.88	786812.25	3956.08	23658	3604.540	87.83	37.9
cap104	1039200.12	815847.11	3104.22	25152	3600.000	88.13	30.4
cap111	946987.60	834992.55	1168.67	6387	3600.250	149.02	19.6
cap112	1033856.56	900192.79	6563.84	7415	3600.020	159.75	12.2
cap113	1106726.50	965955.87	7254.60	8453	3600.010	188.98	9.13
cap114	1205264.29	1064197.88	1347.24	9197	3600.100	213.66	7.11
cap121	903274.84	741896.92	1122.06	12101	3600.000	101.61	9.71
cap122	988021.02	767930.93	7589.72	11528	3600.210	108.25	10.1
cap123	1039819.10	793377.06	4625.81	12336	3600.040	106.88	8.5
cap124	1077859.28	827656.82	7214.81	10170	3600.010	123.54	9.3
cap131	930217.57	717573.91	7048.99	19865	3600.120	53.69	16.1
cap132	1006235.73	729149.95	6085.32	21082	3600.000	65.49	15.5
cap133	1038954.25	738775.63	11862.95	19094	3600.080	76.38	16.2
cap134	1066191.32	755844.84	8412.00	22356	3601.800	74.79	13.4
cap81	939728.01	889797.20	1778.66	4795	3647.520	310.55	37.4
cap82	1021401.86	958749.66	3404.09	4371	3600.190	326.75	39.5
cap83	1094970.48	1025495.62	2435.81	4697	3600.010	271.87	36
cap84	1202095.88	1123780.36	3440.96	4923	3600.020	306.49	29.8
cap91	895172.09	779446.00	5216.79	9425	3608.170	165.05	29
cap92	967000.97	804729.47	6657.32	8263	3600.000	178.28	35.3
cap93	1002948.95	836102.23	3655.91	9186	3620.530	170.98	26.6
cap94	1058771.70	875036.98	4766.36	9818	3600.010	179.47	23.4
shmean	1014672.01	838312.70	3719.25	11121	3603.478	134.99	21.3

Table 35: Detailed results for the SCFLP instances with random seed 5 using *LNS check*.

	Primal bound	Dual bound	Primal integral	Nodes	Time (sec)	Iter per node	BC per node
cap101	904749.07	765452.05	5793.84	22320	3600.070	61.00	42.4
cap102	972119.62	778623.45	5213.97	21513	3600.510	87.50	42.4
cap103	1002798.03	784318.44	2895.70	22194	3601.050	88.84	35.1
cap104	1040854.41	811382.23	3853.10	25193	3600.020	85.08	29.3
cap111	942125.93	836759.80	2509.27	7154	3600.010	147.37	16.8
cap112	1025047.66	901911.88	1408.87	8135	3600.000	164.46	9.28
cap113	1096461.29	966922.69	2546.95	8683	3600.010	181.03	8.42
cap114	1205542.77	1064541.31	3492.33	7811	3600.010	228.27	8.96
cap121	903897.09	740024.32	4208.51	10760	3600.010	93.74	11.8
cap122	969236.65	766696.83	4854.92	11797	3600.000	101.43	9.23
cap123	1024747.62	789759.27	4061.89	10551	3600.020	109.00	11.3
cap124	1062200.46	827091.61	4013.72	11258	3600.000	108.46	8.95
cap131	907689.32	716601.31	3612.22	17988	3602.540	53.03	16.4
cap132	1022549.93	729108.46	16165.38	21708	3600.190	64.87	14.9
cap133	1019696.59	738355.64	2253.55	21112	3600.020	71.06	15.9
cap134	1045175.64	755830.18	3757.85	20519	3600.010	71.94	13.3
cap81	942038.35	888468.02	1783.84	4796	3600.010	292.38	41.3
cap82	1018069.16	961954.04	412.20	5002	3600.000	301.02	31
cap83	1097429.98	1021695.15	2708.65	4360	3696.580	280.85	39.8
cap84	1202862.85	1125137.71	1555.00	4612	3600.120	302.22	33.7
cap91	893532.89	778678.73	1299.27	9302	3689.730	152.31	26
cap92	960030.77	806060.01	2056.82	8444	3600.190	164.12	28.6
cap93	1000799.50	834158.18	4279.15	8468	3600.030	180.75	28
cap94	1055552.69	874299.35	890.21	10508	3602.920	177.20	19.4
shmean	1009949.93	837844.20	2832.30	10966	3607.996	130.63	20.5

Table 36: Detailed results for the SCFLP instances with random seed 7 using *LNS check*.

	Primal bound	Dual bound	Primal integral	Nodes	Time (sec)	Iter per node	BC per node
cap101	889890.12	766725.12	2905.57	7293	3602.370	71.76	48.7
cap102	969566.82	775962.98	6449.85	22502	3614.820	81.52	43.5
cap103	1001071.23	788829.37	2515.95	26447	3600.000	82.83	30.7
cap104	1044035.32	815599.63	4803.46	25377	3744.620	87.49	27.3
cap111	949297.60	836613.50	2389.43	6928	3600.020	166.58	15.9
cap112	1036257.89	903411.69	4101.65	7926	3600.050	159.51	11.2
cap113	1116256.31	968666.28	9346.77	7484	3600.020	205.07	9.92
cap114	1217125.76	1065424.91	8397.79	5842	3600.030	252.89	14.1
cap121	893000.57	742878.78	1311.14	11465	3600.000	100.50	9.64
cap122	981413.23	767575.06	10826.09	11737	3600.050	96.82	10.2
cap123	1029013.53	792509.55	12761.47	9833	3600.140	113.59	11.9
cap124	1047211.42	829908.87	9731.68	10705	3600.010	123.11	9.48
cap131	945946.21	719917.62	14461.73	27620	3605.260	52.72	14.6
cap132	991085.00	734917.62	10026.07	23142	3600.010	64.66	15
cap133	1035593.60	741590.60	12771.77	14880	3601.760	81.77	15.7
cap134	1042360.40	765882.82	4610.43	23741	3600.000	70.52	12.5
cap81	942863.80	891962.84	2068.46	4642	3600.000	299.07	44.9
cap82	1018711.90	957562.93	2173.85	4232	3600.010	312.99	42.2
cap83	1093211.26	1029629.68	1735.22	4592	3600.010	293.66	36.6
cap84	1199557.50	1123925.86	2024.85	4882	3600.010	286.72	30.7
cap91	893754.84	777775.28	3994.03	9115	3607.790	149.33	24
cap92	952516.19	807874.06	313.37	9038	3600.010	173.68	28.4
cap93	999108.40	836985.26	1711.08	9260	3600.000	171.04	23.5
cap94	1046024.71	873489.25	8220.05	7935	3600.000	204.73	27
shmean	1010694.61	840110.75	4222.73	10346	3607.262	135.33	21.1

Table 37: Detailed results for the SCFLP instances with random seed 1 using *Trust Region*.

	Primal bound	Dual bound	Primal integral	Nodes	Time (sec)	Iter per node	BC per node
cap101	908040.73	773718.07	4681.00	23611	3600.110	72.62	48.3
cap102	976104.71	784592.13	1830.31	19102	3601.750	87.01	41.2
cap103	1009018.68	802652.17	2912.83	25131	3600.000	86.89	33.4
cap104	1052682.18	836218.07	6554.90	21207	3600.200	84.48	28
cap111	954166.60	844766.74	4736.76	5656	3600.050	187.68	18.9
cap112	1032674.68	910947.10	3970.12	8109	3600.000	185.32	8.96
cap113	1119995.19	977434.25	9805.15	7408	3600.010	205.40	10.4
cap114	1211358.21	1076404.25	873.44	8114	3600.100	233.39	7.33
cap121	908806.95	748464.10	5369.05	11313	3600.110	103.76	9.71
cap122	992073.76	775434.39	9760.55	10582	3600.010	116.31	10.2
cap123	1039441.23	799840.08	4886.17	10925	3600.060	98.02	11.8
cap124	1086193.31	842465.32	6992.76	12167	3600.060	106.34	8.01
cap131	947679.18	727026.17	15249.21	26371	3600.020	50.55	16.4
cap132	1007468.50	742026.17	12558.13	25385	3600.050	69.20	13.4
cap133	1030220.96	752421.64	8475.43	17531	3603.890	77.09	15.8
cap134	1053167.94	769141.45	4335.19	19693	3600.020	74.71	12.2
cap81	947698.99	897324.33	1683.59	3560	3602.730	298.83	44.9
cap82	1029144.28	972932.79	1753.50	4611	3600.010	315.17	34.8
cap83	1099144.28	1034423.85	555.93	5288	3600.250	293.29	28.7
cap84	1207959.05	1132144.19	806.67	5084	3600.010	273.77	29
cap91	896110.28	785822.00	350.20	10204	3600.020	139.74	24.3
cap92	977983.31	814680.79	6309.15	7541	3600.000	164.86	33.5
cap93	1018916.20	839935.35	6620.42	8337	3615.360	179.70	25.9
cap94	1069418.50	886954.11	6273.82	8057	3600.000	190.06	25.1
shmean	1020856.34	848949.67	3760.78	10743	3601.033	135.53	20.3

Table 38: Detailed results for the SCFLP instances with random seed 2 using *Trust Region*.

	Primal bound	Dual bound	Primal integral	Nodes	Time (sec)	Iter per node	BC per node
cap101	905822.96	772812.67	5101.41	23560	3600.040	71.74	52.3
cap102	977418.20	784006.54	5361.62	21856	3607.090	88.52	38.1
cap103	1004310.86	791325.62	2774.13	21373	3600.080	90.09	32.8
cap104	1040620.71	817233.07	3995.77	18181	3600.060	92.14	29.5
cap111	960783.23	843749.79	3909.72	5782	3600.240	187.53	21.2
cap112	1040185.60	910233.83	5566.05	7337	3600.160	213.51	10.2
cap113	1112035.04	976353.70	6599.53	9611	3600.040	203.98	6.8
cap114	1203918.92	1075721.31	994.07	12063	3600.060	216.50	4.82
cap121	914419.08	747142.75	10663.57	13310	3600.080	94.24	8.6
cap122	1002315.63	774819.42	14973.57	10860	3600.040	106.58	11.4
cap123	1043613.57	797195.92	9672.45	12954	3600.130	100.48	8.37
cap124	1070739.52	836850.16	9087.43	11369	3600.020	116.33	8.11
cap131	947020.28	724774.10	4713.65	26661	3602.390	51.58	17
cap132	1014501.30	738498.34	11977.97	22369	3600.030	75.08	16.1
cap133	1047285.21	745738.69	14571.49	16392	3603.580	76.89	14.9
cap134	1082167.66	766226.79	14695.82	23879	3601.630	80.52	11
cap81	948626.17	899789.42	1628.09	5174	3600.270	289.13	37.9
cap82	1032510.18	968199.83	3003.41	4823	3600.390	304.04	35.7
cap83	1103049.14	1045171.18	903.90	6033	3600.000	306.85	23.5
cap84	1212484.71	1147963.77	1567.56	6507	3600.000	280.07	20.9
cap91	898547.49	790848.36	3124.52	9522	3653.460	167.58	27.8
cap92	968376.81	812512.96	3086.58	7504	3696.980	171.77	33.7
cap93	1018432.61	837264.66	6637.57	7811	3600.340	179.42	28.9
cap94	1052763.70	882547.94	1899.67	9231	3600.000	185.90	21.3
shmean	1022036.19	846888.42	4588.43	11431	3606.899	138.26	19.4

Table 39: Detailed results for the SCFLP instances with random seed 3 using *Trust Region*.

	Primal bound	Dual bound	Primal integral	Nodes	Time (sec)	Iter per node	BC per node
cap101	897435.33	767855.28	1655.19	21685	3600.040	83.69	49.8
cap102	969933.44	778873.02	5428.26	20560	3600.030	80.48	39.8
cap103	1005613.93	797517.65	2719.09	23968	3606.390	90.10	35.6
cap104	1053859.91	816723.36	7668.55	21546	3600.040	90.58	30.3
cap111	950143.45	835138.79	1921.95	6930	3600.050	173.77	16.1
cap112	1018742.92	901068.36	4657.51	8158	3600.010	181.08	9.48
cap113	1104715.04	966582.03	6896.40	7934	3600.010	201.81	9.57
cap114	1205264.29	1066242.86	1324.77	9006	3600.000	239.78	7.48
cap121	919994.98	744458.27	7154.70	13528	3600.060	98.63	9.29
cap122	970116.22	771507.61	1580.29	13911	3600.000	104.76	8.07
cap123	1033080.87	795142.63	11980.28	13701	3600.010	108.29	7.97
cap124	1083869.24	834293.64	9004.50	11200	3600.140	121.58	8.85
cap131	916274.43	721452.95	1044.73	28381	3600.060	55.50	13
cap132	1004356.43	734164.99	5259.24	19984	3600.080	71.50	15
cap133	1018331.20	747346.32	1924.07	21463	3604.210	78.13	15.7
cap134	1104291.26	763430.22	25408.01	18099	3603.150	85.90	12.7
cap81	939649.41	892266.26	1959.72	5344	3600.010	311.32	32.6
cap82	1023286.78	957776.70	3141.20	4928	3600.020	302.41	37.1
cap83	1099366.19	1028816.91	3006.08	4342	3600.000	338.30	37.7
cap84	1201265.00	1125506.79	3249.79	4973	3600.070	290.52	29.9
cap91	882704.25	779961.72	243.99	10006	3600.000	149.86	24.5
cap92	963706.56	810080.96	5566.81	8114	3600.930	179.37	30.9
cap93	1005205.11	836764.00	2893.13	9172	3601.590	171.66	24.2
cap94	1049995.80	875559.87	3292.43	9487	3600.060	192.31	21.6
shmean	1014173.98	841584.57	3405.99	11413	3600.706	139.77	19.7

Table 40: Detailed results for the SCFLP instances with random seed 5 using *Trust Region*.

	Primal bound	Dual bound	Primal integral	Nodes	Time (sec)	Iter per node	BC per node
cap101	894769.08	769716.56	2039.51	18920	3600.010	64.49	34.5
cap102	972611.89	775956.09	5778.62	20503	3600.180	89.08	43.6
cap103	1005055.60	789532.97	2795.02	20379	3600.070	82.33	33.4
cap104	1035634.52	819180.75	5373.64	17781	3600.000	90.57	29.6
cap111	953038.79	836804.51	6484.76	7192	3600.280	159.41	15
cap112	1037303.70	903823.06	4796.26	6620	3600.190	216.64	13.9
cap113	1104817.56	967956.88	5183.51	7082	3600.130	208.61	11.7
cap114	1212333.19	1065274.18	7012.81	5905	3600.020	240.24	13
cap121	904831.25	742668.52	3229.74	15166	3600.010	89.84	7.6
cap122	973693.75	767745.08	5038.91	13795	3600.060	107.42	7.67
cap123	1032138.78	792151.06	6734.92	11881	3600.120	112.80	9.17
cap124	1081907.81	830176.78	11046.34	10460	3600.010	124.96	9.08
cap131	938480.65	720841.07	12586.54	24481	3600.030	63.89	15.5
cap132	988561.31	735078.56	6854.83	18521	3600.090	67.61	14.9
cap133	1018420.11	742308.18	4525.57	13336	3600.060	81.60	15.4
cap134	1061706.60	769630.13	11429.89	24915	3600.020	72.65	11.9
cap81	940882.11	891542.13	625.50	5146	3600.010	293.66	36.9
cap82	1023491.75	958386.43	2731.33	4962	3600.000	275.74	37
cap83	1097862.85	1025653.72	2422.51	5012	3600.000	285.21	32.6
cap84	1202405.68	1125501.90	2532.27	4512	3600.000	312.09	33.7
cap91	893532.89	780026.27	1275.24	10303	3607.060	162.36	24.7
cap92	956393.83	808042.92	2248.16	8433	3605.600	174.71	27
cap93	1014632.45	835306.09	9844.33	8269	3600.010	179.26	26.9
cap94	1063962.25	878895.89	5490.65	8452	3628.330	201.10	25.3
shmean	1013761.23	840856.90	4349.55	10579	3601.757	138.27	20.2

Table 41: Detailed results for the SCFLP instances with random seed 7 using *Trust Region*.

	Primal bound	Dual bound	Primal integral	Nodes	Time (sec)	Iter per node	BC per node
cap101	898511.02	766725.12	3645.78	22043	3600.030	70.70	46.9
cap102	967983.83	769503.61	6130.26	19264	3600.070	85.01	43.2
cap103	997022.93	786249.82	460.90	24238	3600.030	82.86	31.9
cap104	1043408.71	808208.61	4430.71	20677	3604.510	90.48	30.4
cap111	944425.28	836657.60	598.85	7185	3600.000	155.04	15.2
cap112	1026531.28	902166.36	727.87	7555	3600.010	151.09	12.1
cap113	1105214.37	966845.51	7594.59	6801	3600.000	180.20	11.4
cap114	1206311.43	1065883.03	5307.88	9676	3600.040	195.60	6.56
cap121	893000.62	740489.76	1048.08	12667	3600.040	88.58	9.67
cap122	987133.82	766086.94	12901.56	10058	3600.010	110.27	11.4
cap123	1027678.10	793518.44	9312.45	13298	3600.100	94.11	8.62
cap124	1079688.55	828787.51	13427.36	12863	3600.080	100.12	7.17
cap131	919458.72	715933.33	5176.17	21812	3600.010	53.60	14.7
cap132	1001084.01	729581.32	12191.47	23314	3600.360	65.31	14.5
cap133	1040743.16	740657.36	10474.82	18290	3600.010	72.83	14.9
cap134	1059899.84	754788.10	11937.39	20217	3600.100	76.95	13.2
cap81	942566.74	892497.38	2030.94	4501	3600.010	303.08	44
cap82	1022537.13	956917.70	2230.57	5039	3600.000	281.53	36.7
cap83	1095568.92	1033806.42	2031.40	4901	3600.090	280.08	35.8
cap84	1196119.38	1132841.75	552.00	5906	3600.010	272.47	22.4
cap91	896299.04	776646.59	5094.95	10432	3723.960	134.73	25.1
cap92	952516.19	806641.68	314.90	9305	3600.040	159.46	27.6
cap93	998051.85	832628.14	4997.56	8493	3600.010	168.52	29.2
cap94	1044321.96	873934.26	4075.77	9610	3600.010	175.54	22.4
shmean	1011163.19	838248.46	3331.24	11232	3605.314	127.59	20.1

Table 42: Detailed results for the SCFLP instances with random seed 1 using *TR LNS check*.

	Primal bound	Dual bound	Primal integral	Nodes	Time (sec)	Iter per node	BC per node
cap101	903265.17	771134.84	3160.38	21317	3600.010	74.49	50.8
cap102	988723.84	776201.51	6318.47	19393	3602.700	85.84	43.7
cap103	1020565.73	794183.06	6706.11	22252	3652.310	92.41	35.2
cap104	1045069.18	816204.87	6028.40	16402	3602.370	86.38	29.5
cap111	951351.61	844722.04	1155.57	5605	3600.030	150.54	22.4
cap112	1037974.80	910427.44	6708.29	6563	3600.000	184.36	12.6
cap113	1097938.59	977427.53	3014.16	9092	3600.060	178.90	7.42
cap114	1220230.45	1074868.48	3199.05	9413	3600.020	215.13	6.4
cap121	901771.44	746749.96	2869.15	11041	3600.040	82.49	10.3
cap122	967324.29	773469.48	1420.78	10396	3600.090	89.22	11
cap123	1030225.10	799890.52	1087.05	13367	3600.010	100.51	7.99
cap124	1072381.70	835539.96	2087.47	12173	3600.010	97.62	7.48
cap131	928127.66	723288.95	9503.60	18713	3600.040	53.67	14.8
cap132	1012756.93	735437.06	13076.03	17594	3601.940	65.81	15.7
cap133	1068933.37	745394.59	16273.84	19622	3600.070	76.14	16.4
cap134	1069392.72	767405.09	6077.65	23363	3600.060	74.66	12.4
cap81	945336.05	899784.66	1260.54	4291	3600.010	301.66	49.5
cap82	1025389.36	965875.38	1153.42	4241	3600.090	290.98	34.7
cap83	1101968.79	1032631.70	1978.99	4821	3600.010	290.19	32.9
cap84	1209480.57	1131732.39	3040.76	4873	3600.000	282.64	32.3
cap91	895788.80	784941.79	233.12	10497	3603.830	127.11	25.1
cap92	978542.31	812826.15	4938.90	8217	3600.000	167.43	33.2
cap93	1014469.48	838839.25	5341.41	7890	3615.870	176.31	30.4
cap94	1066928.66	879496.81	4203.13	7936	3600.030	193.51	27.8
shmean	1019762.62	845084.72	3282.59	10483	3603.301	129.78	21.1

Table 43: Detailed results for the SCFLP instances with random seed 2 using *TR LNS check*.

	Primal bound	Dual bound	Primal integral	Nodes	Time (sec)	Iter per node	BC per node
cap101	905822.96	771539.45	5122.88	23464	3616.190	76.87	52.5
cap102	978670.80	774106.41	5414.56	23096	3600.110	89.29	40.9
cap103	1012834.04	790815.84	3964.97	23536	3605.730	88.77	34.1
cap104	1040620.71	813489.61	3525.80	22508	3600.020	84.40	28
cap111	953187.83	843642.98	1294.58	6546	3600.010	167.59	16.7
cap112	1033117.74	910998.27	3654.87	8823	3600.000	175.46	9.22
cap113	1108615.45	977293.06	5464.13	8655	3600.000	197.54	7.9
cap114	1215172.98	1073776.51	5379.86	9111	3600.000	211.12	6.96
cap121	889130.30	745123.41	2248.28	9003	3600.020	98.95	11.9
cap122	965660.13	772707.76	2777.44	13904	3600.090	87.82	7.34
cap123	1019013.76	797013.97	2998.44	11511	3600.070	97.66	9.44
cap124	1063147.70	833817.88	3816.41	12330	3600.020	99.80	7.68
cap131	947020.29	721129.87	4736.03	22739	3612.100	53.67	15
cap132	1014501.30	732353.83	11988.63	20981	3602.960	67.09	17.3
cap133	1032710.01	744056.32	8674.61	20327	3600.060	74.93	14.9
cap134	1069294.82	760209.24	12977.02	19231	3600.040	82.69	12
cap81	947669.47	903159.28	875.93	5406	3600.010	293.08	34.8
cap82	1026058.39	970893.07	1181.80	5190	3600.000	297.42	29.1
cap83	1101164.34	1041175.68	634.11	6524	3600.000	276.39	23.2
cap84	1208049.14	1148324.75	253.26	7833	3600.020	276.09	15.4
cap91	891477.10	782221.22	250.10	8853	3601.680	148.06	26.5
cap92	963801.84	808743.73	1888.22	8102	3600.200	160.16	31.1
cap93	1006098.16	842887.56	2504.10	9154	3600.000	188.71	23.3
cap94	1056787.57	878589.78	3573.24	9931	3600.010	169.66	19.9
shmean	1015469.11	844668.96	2747.17	11669	3601.637	131.76	18.6

Table 44: Detailed results for the SCFLP instances with random seed 3 using *TR LNS check*.

	Primal bound	Dual bound	Primal integral	Nodes	Time (sec)	Iter per node	BC per node
cap101	899002.07	760383.17	2236.90	21773	3600.010	76.85	54.7
cap102	968023.90	775242.55	4402.56	23372	3600.050	86.97	39.3
cap103	1009200.12	784401.45	6236.09	16535	3604.050	92.45	40.4
cap104	1036381.06	814640.42	2735.17	17949	3600.000	86.62	30.6
cap111	949519.49	834991.67	1852.57	6347	3600.010	148.15	20.4
cap112	1033856.56	899854.30	6560.32	6995	3600.000	164.56	12.9
cap113	1106726.50	966151.15	7245.04	7829	3600.110	206.41	9.96
cap114	1202701.46	1065047.75	964.83	8571	3600.010	229.37	7.54
cap121	906886.54	743112.78	4866.29	14056	3600.470	78.34	7.86
cap122	988021.02	768117.02	7677.54	11650	3600.010	107.01	9.8
cap123	1030586.88	793702.75	1809.98	10842	3600.150	109.85	9.51
cap124	1067741.74	832960.32	5257.70	12262	3600.040	100.97	7.08
cap131	930217.57	717079.61	7137.64	17383	3603.350	53.05	15.8
cap132	998314.16	728208.75	2962.01	17053	3603.510	66.09	16.2
cap133	1040242.22	737194.38	11608.26	13696	3600.030	77.02	16.6
cap134	1094305.14	754654.31	20951.42	15305	3602.350	78.94	13.4
cap81	936456.65	887711.76	832.66	5041	3600.000	299.99	35.7
cap82	1022272.97	956176.76	2966.35	5002	3600.010	288.05	37.7
cap83	1091006.12	1026997.34	986.19	5110	3600.040	285.50	30.4
cap84	1194430.48	1122881.05	2464.13	4910	3600.010	307.61	28.3
cap91	885448.79	779492.18	1359.46	10685	3600.080	148.12	23.7
cap92	949955.68	809555.52	980.39	9948	3600.010	156.53	26.5
cap93	1003524.94	836042.67	3363.24	9420	3633.220	170.06	24.4
cap94	1048459.31	875077.71	4651.94	9061	3600.000	177.81	25.8
shmean	1013161.67	838108.68	3358.55	10536	3601.974	132.23	20.4

Table 45: Detailed results for the SCFLP instances with random seed 5 using *TR LNS check*.

	Primal bound	Dual bound	Primal integral	Nodes	Time (sec)	Iter per node	BC per node
cap101	893313.36	766593.43	1569.54	21653	3600.050	57.49	37.1
cap102	970323.65	768397.80	7262.28	15677	3600.060	89.86	42.3
cap103	1003751.84	792966.05	2796.86	21553	3600.010	87.43	37.4
cap104	1033751.84	811086.21	1875.25	20585	3600.070	85.42	28.8
cap111	942125.93	836751.95	2541.18	7076	3600.310	147.26	16.7
cap112	1025047.66	901903.72	1535.95	8018	3600.010	168.11	8.92
cap113	1096461.29	966819.40	2670.64	8651	3600.030	177.81	8.6
cap114	1202254.49	1065382.58	3204.31	8157	3600.030	231.89	7.85
cap121	906787.08	740983.88	4022.09	10563	3600.140	97.82	11.6
cap122	963505.01	766814.92	1799.41	11711	3600.720	98.85	9.79
cap123	1032274.81	788727.28	12321.11	7483	3601.690	119.13	16.4
cap124	1082232.32	826719.50	11054.30	9709	3600.020	106.17	10.6
cap131	907223.75	716067.49	1700.88	16994	3600.210	55.33	15
cap132	1015515.78	730956.92	13560.61	17639	3603.340	63.23	16.4
cap133	1029108.31	748058.09	6876.81	20863	3602.470	69.43	15.2
cap134	1086246.68	754741.92	19216.52	17718	3600.100	79.23	13.3
cap81	939922.98	888991.59	815.05	4525	3600.140	289.52	43.8
cap82	1018069.16	958484.95	822.78	5397	3600.010	263.25	32.4
cap83	1091525.05	1029234.72	1090.55	5085	3600.020	270.97	29.1
cap84	1200974.98	1130271.98	1178.29	5387	3600.230	290.04	28.7
cap91	893532.89	778389.66	1273.76	9124	3600.260	156.86	29.5
cap92	961390.15	806529.44	2914.06	8298	3610.740	177.65	30.5
cap93	993608.73	831532.91	3419.62	8432	3600.000	165.15	27.3
cap94	1057849.13	879817.20	2681.71	9885	3600.000	176.14	21.5
shmean	1011109.68	838712.52	2977.59	10396	3600.860	130.25	20.6

Table 46: Detailed results for the SCFLP instances with random seed 7 using *TR LNS check*.