OPEN ACCESS

King's Research Portal

DOI:
10.1051/ro/2021148

Document Version
Peer reviewed version

Link to publication record in King's Research Portal

Citation for published version (APA):
M'Halla Ep Aounallah, R., & Ben Nejma, I. (2021). A beam search for the equality generalized symmetric traveling salesman problem. *RAIRO - Operations Research*, *55*(5), 3021-3039. https://doi.org/10.1051/ro/2021148

Citing this paper
Please note that where the full-text provided on King's Research Portal is the Author Accepted Manuscript or Post-Print version this may differ from the final Published version. If citing, it is advised that you check and use the publisher's definitive version for pagination, volume/issue, and date of publication details. And where the final published version is provided on the Research Portal, if citing you are again advised to check the publisher's website for any subsequent corrections.

General rights
Copyright and moral rights for the publications made accessible in the Research Portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognize and abide by the legal requirements associated with these rights.

•Users may download and print one copy of any publication from the Research Portal for the purpose of private study or research.
•You may not further distribute the material or use it for any profit-making activity or commercial gain
•You may freely distribute the URL identifying the publication in the Research Portal

Take down policy
If you believe that this document breaches copyright please contact librarypure@kcl.ac.uk providing details, and we will remove access to the work immediately and investigate your claim.

Download date: 24. Oct. 2024

A beam search for the equality generalized symmetric traveling salesman problem

Ibtissem Ben Nejma

Logistics Department, Institut Superieur du Transport et de la Logistique de Sousse, University of Sousse, Sousse, Tunisia.

Rym M'Hallah

Engineering Department, Kings College London, Strand, London WC2R 2LS, UK.

Abstract

This paper studies the equality generalized symmetric traveling salesman problem (EGSTSP). A salesman has to visit a predefined set of countries. S/he must determine exactly one city (of a subset of cities) to visit in each country and the sequence of the countries such that s/he minimizes the overall travel cost. From an academic perspective, EGSTSP is very important. It is NP-hard. Its relaxed version TSP is itself NP-hard, and no exact technique solves large difficult instances. From a logistic perspective, EGSTSP has a broad range of applications that vary from sea, air, and train shipping to emergency relief to elections and polling to airlines' scheduling to urban transportation. During the COVID-19 pandemic, the roll-out of vaccines further emphasizes the importance of this problem. Pharmaceutical firms are challenged not only by a viable production schedule but also by a flawless distribution plan especially that some of these vaccines must be stored at extremely low temperatures. This paper proposes an approximate tree-based search technique for EGSTSP. It uses a beam search with low and high level hybridization. The low-level hybridization applies a swap based local search to each partial solution of a node of a tree whereas the high-level hybridization applies 2-Opt, 3-Opt or Lin-Kernighan to the incumbent. Empirical results provide computational evidence that the proposed approach solves large instances with 89 countries and 442 cities in few seconds while matching the best known cost of 8 out of 36 instances and being less than 1.78% away from the best known solution for 27 instances.

Email addresses: `nejmalog@gmail.com` (Ibtissem Ben Nejma), `rym.mhallah@kcl.ac.uk` (Rym M'Hallah)

*Corresponding author. Tel: +44-749-8063841.

Preprint submitted to RAIRO - Operations Research *September 20, 2021*

This provisional PDF is the accepted version. The article should be cited as: RAIRO: RO, doi: 10.1051/ro/2021148

Keywords: Generalized traveling salesman, beam search, symmetric traveling salesman, k-opt, Lin-Kernighan heuristic.

1. Introduction

Logistics in general and transportation in particular are the cornerstones of modern life. Their importance emanates from their multi-fold repercussions on the cost of goods, profit margins of transportation companies, clients' service quality, drivers' well being, and air pollution. In fact, they involve several parties: end clients, manufacturers, distributors, drivers, stock holders, etc. In addition, they require the scheduling of several interrelated tasks that are dynamic in nature and constrained in time and space. The economic and temporal constraints augment their complexity. Solving them requires the migration of tools from diverse disciplines including information technology, optimization, and vehicle routing.

Among the most widely studied transportation problems is the traveling salesman problem (TSP). A traveler has to visit a finite number of countries starting from one country and returning back to it, and visiting every country exactly once. The objective is to find a minimal cost route, where the cost can be total duration, travel distance, etc. TSP's importance emanates from its occurrence as a subproblem of complex real life problems in the transport of passengers/goods and in scheduling. For these problems, TSP identifies a minimal-cost itinerary for each salesman. For example, TSP is a special case of the equality generalized symmetric traveling salesman problem (EGSTSP), where the salesman chooses exactly one of many cities of a country to visit; i.e., a TSP with a covering constraint.

Formally, consider a set of nodes that are divided into clusters. EGSTSP searches for the shortest route that visits exactly one node from every cluster starting and ending at the same cluster. EGSTSP is more difficult than TSP because of the combinatorial aspect added by the sizes of the clusters. EGSTSP occurs in several real-life applications such as maritime ship routing, distribution of medical supplies, urban waste management, telecommunication networks, logistics, rapid post dispatching, VLSI, circuit designs, and in laser cutting to determine the trajectory of a laster cutter [18, 21]. During the COVID-19 pandemic, EGSTSP has drawn a lot of attention. With reduced air-traffic and disrupted logistic chains, the procurement and dispatching of goods to confined customers and isolated cities has become a true challenge. In addition, the availability of a vaccine raises the issue of its fair distribution and of health care equity. Some of these vaccines impose a cold chain that can't be broken. In such cases, optimizing the distribution plan is of prime importance. This optimization is equivalent to solving a large scale EGSTSP. Evidently, its exact solution may be challenging. However, the continuous advancement of the computing technologies provides near-optimal solutions to such difficult problems. They are allowing approximate methods to undertake a more extensive search; thus obtaining nearer-global optima in shorter times.

EGSTSP has been solved by exact approaches (such as dynamic programming, branch and bound, and branch and cut), and by approximate ones such as local improvement heuristics (k-opt, swap, insertion, etc.), and meta-heuristics (tabu search, ant colony, genetic algorithms, etc.). This paper proposes a new approximate hybrid approach for the EGSTSP. Hybrid heuristics have identified the best known solutions to several complex combinatorial optimization problems. They are powerful search methods because they tackle two competing goals: exploration and exploitation. Exploration is a diversification of the search. It investigates the solution space in order to determine the part that has a higher chance of containing the global optimum. Exploitation refines (or intensifies) the search on the part of the space that has a high potential of containing the global optimum.

The proposed hybrid heuristic is a beam search (BS) (i.e., a truncated branch and bound) that is augmented with improvement techniques. It ensures exploration via a standard width-first BS and exploitation via local search heuristics. BS strives for global optimization while local search heuristics strive for local optimization in the global optimum's neighborhood. That is, BS can be assimilated to evolution while local search to learning. Generally, synchronization of evolution and learning yields efficient hybrid heuristics. Specifically, the proposed hybrid BS embeds

- a low-level hybridization, which addresses the functional composition of BS by subjecting the partial solution at each node to a local search; and
- a high level hybridization that maintains BS self containing by subjecting the incumbent of BS to a k-opt type of search.

To the best of the authors' knowledge, this is the first application of BS to EGSTSP. In addition, the hybridization explores the success of local search to assess the nodes of the tree and to estimate their potential. It subsequently chooses the nodes with the best potential to branch on and prunes the non-promising ones; thus, it explores the search space's parts that contain near-global optima while it discards the others. It then applies a 2-opt, a 3-opt or the notorious improved Lin-Kernighan (LK) heuristic [9] to its incumbent. The computational investigation provides computational evidence of the good performance of the hybridized BS within a reduced runtime. Its deviation from the best known solution is less than 0.0578% for half of the instances and less than 1.78% for three quarters of them.

Section 2 defines the problem. Section 3 reviews the literature on EGSTSP. Section 4 details the proposed approaches. It presents the algorithm of a standard BS, its adaptation to EGSTSP, the low-level hybridized BS, and the high-level hybridized BS when applying each of the three local improvement methods: 2-Opt, 3-Opt and LK. Section 5 presents the computational results, which assess the efficiency of the methods in terms of solution quality and runtime and highlights the utility of the hybridizations. Finally, Section 6 summarizes the findings and gives potential research extensions.

2. Problem Definition

EGSTSP is an NP-hard combinatorial optimization problem. It consists of finding the optimal path of a salesman who has to travel through a set of countries while visiting exactly one city from each country and visiting every country once. The optimal path minimizes the total traveled cost. Hence, the salesman must determine for each country the city s/he will visit and the order of visit of the countries. EGSTSP is more complex than TSP. For TSP, each country consists of a single town while EGSTSP has the additional complexity of choosing a city from each country. Because it extends TSP, which is NP-hard, EGSTSP is also NP-hard.

Herein, we define EGSTSP using the notation of [6, 7] and present their integer linear program (ILP). Formally, consider a complete non-oriented graph $G = (N, E)$ where $N = \{1, \ldots, n\}$ is a set of nodes that are divided into m mutually exclusive clusters $C_h,\ h = 1, \ldots, m$, and $m \geq 3$. $E = \{[i, j] : i \in N, j \in N, i \neq j\}$ denotes the set of edges e connecting pairs (i, j) of distinct nodes $i \in N$ and $j \in N,\ i \neq j$. The cost of traveling through edge $e \in E$ is d_e. This cost may be assimilated to a linear function of the Euclidean distance between i and j. The objective of EGSTSP is to determine a minimal cost cycle $T \subseteq E$ such that T includes exactly one city from each cluster, and each cluster is visited once.

To define the ILP model of EGSTSP, we introduce the following notation. For a subset $S \subseteq N,\ E(S) := \{[i, j] \in E : i \in S,\ j \in S\}$ denotes the set of edges with both endnodes in S and $\delta(S) := \{[i, j] \in E : i \in S,\ j \notin S\}$ the set of edges with exactly one end node in S. For simplicity, we denote $\delta(\{v\})$ by $\delta(v)$, for $v \in N$.

ILP uses two types of binary variables: $x_e = 1$ if the salesman travels through edge $e \in E$ and 0 otherwise, and $y_v = 1$ if the salesman visits node

$v \in N$ and 0 otherwise. Using the aforementioned notation and these two sets of binary variables, EGSTSP can be formulated as follows.

$$\min z = \sum_{e \in E} d_e x_e \tag{1}$$

$$\text{s.t.} \sum_{e \in \delta(v)} x_e \leq 2y_v \qquad v \in N \tag{2}$$

$$\sum_{v \in C_k} y_v = 1 \qquad k = 1, \ldots, m \tag{3}$$

$$\sum_{e \in \delta(S)} x_e \geq 2(y_i + y_j - 1) \qquad S \subset N, 2 \leq |S| \leq n-2, i \in S, j \in N \setminus S \tag{4}$$

$$x_e \in \{0, 1\} \qquad e \in E \tag{5}$$

$$y_v \in \{0, 1\} \qquad v \in N \tag{6}$$

The objective function, given by Equation (1), minimises the total travel cost. Equation (2) preserves the flow through every node. A node is visited if it has both a predecessor and a successor node; therefore the righthand side is 2; otherwise, the righthand side must be zero. Equation (3) ensures that the tour includes exactly one city from each cluster. Equation (4) guarantees the connectivity of the solution: Each cut separating two visited nodes i and j must be crossed at least twice. Finally, Equations (5) and (6) determine the nature of the decision variables.

Because EGSTSP is NP-hard, solving large instances of EGSTSP using ILP is difficult. Herein, we are interested in efficiently solving large instances of EGSTSP using heuristic methods, and in comparing the heuristics' solutions to the ILP results that are readily available in the literature (and given by z^{lit} in the computational section and in the Appendix).

3. Literature Review

Small instances of the equality generalized TSP (EGTSP) were solved exactly using dynamic programming [26], branch and bound [14], and branch and cut [6]. Large instances have been tackled approximately; for example, Noon and Bean [20] applied the TSP's closest neighborhood heuristic. Lien et al. [16] assimilated EGTSP to a TSP whose number of nodes is three times as large as the number of clusters. Dimitrijevic and Saric [5] developed an alternative transformation that had fewer nodes; i.e., using twice as many nodes as the number of clusters of the original EGTSP. Ben-Arieh et al. [2] opted for a transformation that had as many nodes as the number of clusters of EGTSP using the 'exact' Noon–Bean, and two modifications of the non-exact Fischetti–Salazar–Toth transformation. Helsgaun [9] transformed EGSTSP into a classic TSP and applied LK to the transformed TSP. Karapetyan and Gutin [11] proposed an LK heuristic for EGSTSP. Smith and Imeson [25] applied an iterative remove and insert heuristic for EGSTSP. They opted for three insertion mechanisms: the furthest node, the cheapest, and random insertion. Karapetyan and Gutin [12] also designed a large neighborhood search for EGSTSP. Renaud et al. [22] proposed an Initialization, Insertion and Improvement heuristic that Renaud and Boctor [23] further generalized. Khachai and Neznakhina [13] developed a dynamic programming based heuristic for EGSTSP.

Another surge of the EGSTSP literature came from hybrid approaches. Ardalan et al. [1] hybridized the Imperialist Competitive Algorithm with a local search. Lawrence and Daskin [15] hybridized a random key genetic algorithm with a local search. Their algorithm is quite fast. It identifies its best solution within the first two or three iterations. Its good performance is due to the utility of the local search in identifying the best solution. However, their algorithm is outperformed by the mimetic algorithm of Gutin et al. [8], who combined the advantages of genetic algorithms and local search. Chira et al. [4] designed a "sensible" ant colony system that makes the ants sensitive to the pheromone level in their trail; thus, explore the most promising regions of the search space. Yang et al. [28] augmented ant colony optimization to EGSTSP with a mutation mechanism and a local search. They showed the importance of the local search, in particular, for instances with fewer than 200 nodes. Bontoux et al. [3] proposed a mimetic algorithm whose crossover operator is based on a large neighborhood search.

Different variants of EGSTSP have appeared recently. Sundar and Rathinam [27] applied a branch and cut and Zhou and Brian [30] extended Christofide's TSP algorithm to the multi-depot EGSTSP where there are several travelers; each departing from a different depot (node). Mestria [19] considered the clustered traveling salesman problem, where all nodes of a cluster must be visited in a contiguous manner. The author hybridized a variable neighborhood random descent with local search (for intensification) and with a greedy randomized adaptive search (for diversification). This latter consists of a constructive heuristic and a perturbation method. The author applied several variable neighborhood structures, in a random order. Jian et al. [10] proposed a hybrid genetic ant colony algorithm for the multiple TSP, where each salesman departs from a specific depot and returns to it. Yuan et al. [29] studied the generalized TSP with time windows, where arrival to a city must occur within a time window. They proposed two integer linear programs and valid inequalities that are separated dynamically within a branch-and-cut algorithm. They initiated their branch and bound from a feasible solution built via a simple heuristic. They solved instances with up to 30 clusters within a one-hour runtime. Salman et al. [24] imposed precedence constraints on EGSTSP, developed a new branching rule, and adapted some existing bounds to the problem.

This literature review suggests that EGSTSP was never tackled via BS. It further suggests that hybridization is a key factor in the success of most approaches to TSP related problems. To explore these findings, this paper proposes a hybrid BS that employs local search at each node and applies a k-opt type of search to the incumbent.

4. Proposed Approaches

We efficiently solve EGSTSP using hybridized BS-based algorithms. BS is a truncated tree search. It avoids exhaustive enumeration by branching on a subset of elite nodes, believed to lead to the optimum. They usually have minimal fitness values, which are either the cost of their partial solutions or their upper bounds. At each iteration, ω nodes are selected for branching, where ω is the beam width. The other nodes are permanently discarded, and no backtracking is performed. We enhance the performance of BS by hybridizing it at two levels. The low-level hybridization adds a local search phase at each node of the BS tree. The high-level hybridization applies 2-opt, 3-opt or LK heuristics to the best solution that BS obtains. Section 4.1 describes a standard BS. Section 4.2 explains our adaptation of BS to EGSTSP. Sections 4.3 and 4.4 present the low- and high-level hybridization.

4.1. Standard Beam Search

The pseudo code of a standard BS is given in Figure 1. It consists of an initialization step, an iterative step and a stopping criterion. The initialization step declares the set $\mathcal{N}$ of current nodes of the tree to the root node μ_0 and the set $\mathcal{M}$ of offspring nodes to the empty set. When an initial feasible solution $\mathbf{x}$ is available, this step further sets the incumbent $\mathbf{x}^*$ and its value $z^* = z(\mathbf{x}^*)$ to, respectively, this initial solution $\mathbf{x}$ and its objective function value. When an initial feasible solution is not available, the upper bound z^* is set to ∞.

The iterative step chooses a node from $\mathcal{N}$, and sets it as the current node. It branches out of the current node, and adds all new nodes to $\mathcal{M}$ except for leaves. Leaves constitute feasible solutions; thus, are candidate solutions. A leaf becomes the incumbent whenever its cost is less than z^*.

Initialization

Set $\mathcal{N} = \{\mu_0\}, \ \mathcal{M} = \emptyset$.

If an initial feasible solution $\mathbf{x}$ whose cost $z(\mathbf{x})$ is available, set the incumbent $\mathbf{x}^* = \mathbf{x}$, and its objective function value $z^* = z(\mathbf{x})$; otherwise, set $z^* = \inf$.

Iterative step

1. Choose a node μ of $\mathcal{N}$; branch out μ; and insert the created nodes (i.e., the offsprings of μ) into $\mathcal{M}$.
2. If a node μ of $\mathcal{M}$ is a leaf, then
   - compute its objective function value z_μ;
   - if $z_\mu < z^*$, update z^* and the incumbent solution;
   - remove μ from $\mathcal{M}$.
3. Assess the potential of each node of $\mathcal{M}$.
4. Rank the nodes of $\mathcal{M}$ in a non-descending order of their values.
5. Insert the $\min\{\delta, |\mathcal{M}|\}$ best nodes of $\mathcal{M}$ into $\mathcal{N}$; and set $\mathcal{M} = \emptyset$.

Stopping condition

If $\mathcal{N} = \emptyset$, stop; otherwise, goto the *iterative step*.

Figure 1: A standard BS.

The iterative step appends the ω smallest-cost nodes of $\mathcal{M}$ to $\mathcal{N}$ and re-initializes $\mathcal{M}$ to the empty set. This process is reiterated until no further branching is possible; that is, till $\mathcal{N} = \emptyset$. When applying a width-first BS, the nodes of $\mathcal{N}$ belong to the same level of the tree.

4.2. Proposed Beam Search

This section presents our proposed BS-based method BS_0 for EGSTSP. BS_0 identifies a least cost ordering of the clusters. It assimilates the nodes of the tree to partial solutions (i.e., ordered subsets of C), and branching out of a node to augmenting it with an additional cluster. Its tree starts at the root node (i.e., level $\ell = 0$) with an empty tour, and has at most m levels. A partial solution s^ℓ corresponding to a node at level $\ell, \ell = 1, \ldots, m$, is a sequence of cities $i^1, \ i^2, \cdots, i^\ell$ all belonging to N and to different clusters. As all tree-search techniques, BS_0 has three major steps: branching, assessment, and selection.

The branching of a node of the tree corresponds to appending a cluster to the partial solution of that node. That is, out of a node of level ℓ emanate $m-\ell$ branches; each leading to a different cluster. A node inherits the path of its parent, and appends a cluster to the end of its parent's path. Specifically, branching out of the node corresponding to s^ℓ consists in appending a city from a non-visited cluster to s^ℓ.

The assessment of the cost of a newly created node s^ℓ is based on a straightforward / simple lower bound and on an upper bound. The lower bound is the cost z_{s^ℓ} of the partial solution s^ℓ. It is the sum of the travel costs between the successive nodes of s^ℓ:

$$z_{s^\ell} = d_{i^1,i^2} + d_{i^2,i^3} + \ldots + d_{i^{\ell-2},i^{\ell-1}} + d_{i^{\ell-1},i^\ell}.$$

It is the sum of its parent node's cost $z_{s^{\ell-1}} = d_{i^1,i^2} + d_{i^2,i^3} + \ldots + d_{i^{\ell-2},i^{\ell-1}}$ and of the travel cost $d_{i^{\ell-1},i^\ell}$ from its parent node to the appended cluster. The upper bound is a total-cost of a complete solution constructed by iteratively appending the closest city of a 'not yet assigned' cluster to the partial solution s^ℓ.

At a given level ℓ of the tree, the selection chooses the ω best nodes among all generated child nodes for further branching at the next level $\ell+1$ of the tree. These iterative branching, evaluation and selection processes are repeated until $\ell = m$; that is, until all clusters are visited. Herein, BS_0 is started with a feasible solution obtained via a greedy heuristic that chooses arbitrarily the first city i^1 and iteratively appends the closest city from a non-visited cluster.

In summary, BS_0 is a constructive approach that starts at the root node with an empty tour and appends a cluster at each level of the tree. It stops when the tour has m clusters visited. It has an $O(\omega m)$ worst case time complexity. Thus, our transformation of EGSTSP into TSP is less complex than competing transformations. It maintains $m < n$ nodes whereas TSP considers n nodes.

4.3. Enhanced Beam Search

The low-level hybridized BS, denoted hereafter as BS_1, subjects each partial solution s^ℓ obtained at a node of a level ℓ, $\ell = 3, \ldots, m$, of the tree to a local search. The local search is simple but efficient. It preserves the order of the clusters in s^ℓ but changes the selected node of one or more clusters. It chooses the 'best' city among all nodes of every cluster of the partial solution s^ℓ. At a level $\ell \in \{3, \ldots, m\}$, BS generates $m-\ell$ nodes. Let s^ℓ be one of these nodes and let $s^\ell = ([1], \ldots, [\ell])$, where $[i]$ denotes the ith cluster of the tour. The local search iterates for $h = [2], \ldots, [\ell - 1]$. It fixes the partial paths $[1], \ldots, [h-1]$ and $[h+1], \ldots, [\ell]$, and iterates through all the cities v of cluster C_h. It retains the city $v^* \in C_h$ that minimizes the distance from $[h-1]$ to v to $[h+1]$; i.e.,

$$d_{[h-1]v^*} + d_{v^*[h+1]} = \min_{v \in C_h} \{d_{[h-1]v} + d_{v[h+1]}\}.$$

When applied to a node s^ℓ, the local search has $O(\ell c)$ complexity ($c = \bar{c}$), where $\bar{c} = \max_{h=1,\ldots,m} \{|C_h|\}$ is the maximum number of cities among all clusters. Because it is applied to all $\sum_{\ell=3}^{m} \ell(m - \ell)$ nodes of the tree, the local search increases the complexity of BS_1 to at worst $O(\omega m^2 \bar{c})$. Yet, it allows BS_1 to attenuate the myopic nature of BS_0; i.e., BS_0 may miss the global optimum when it selects the ω best nodes of a level and permanently prunes the others.

4.4. High-Level Hybridized Beam Search

The high level hybridized BS, denoted $BS_\cdot, \cdot = 2, 3, 4$, applies a 2-Opt, a 3-Opt, or LK heuristic to the best solution obtained by BS_1. Because the hybridization is high-level, the worst time complexity of $BS_\cdot, \cdot = 2, 3, 4$ is the sum of the complexity of BS_1 and of the adopted hybridization approach.

The 2-Opt has an $O(m^2)$ complexity where m is the number of clusters of the tour. It chooses two clusters of the tour randomly and reverses the flow between them. It is repeated as long as the solution is improved. For instance, consider a tour $[1], [2], \ldots, [i-1], [i], [i+1], \ldots, [j-1], [j], [j+1], \ldots, [m], [1]$, where $[i]$ denotes the $[i]^{th}$ cluster of the tour. When 2-Opt chooses randomly clusters $[i]$ and $[j]$, it generates the new solution $[1], [2], \ldots, [i-1], [j], [j-1], \ldots, [i+1], [i], [j+1], \ldots, [m], [1]$.

The 3-Opt has an $O(m^3)$ complexity where m is the number of clusters of the tour. For a tour $[1], [2], \ldots, [i-1], [i], [i+1], \ldots, [j-1], [j], [j+1], \ldots, [\kappa-1], [\kappa], [\kappa+1], \ldots, [m], [1]$, 3-Opt chooses randomly three clusters $[i]$, $[j]$ and $[\kappa]$ of the tour, and generates the new solution $[1], [2], \ldots, [i], [\kappa], [\kappa-1], [j+1], [i+1], \ldots, [j], [\kappa+1], \ldots, [m], [1]$. It repeats this process as long as the solution is improved.

LK yields near-global optima when started from a large number of initial solutions. Any perturbation of its best solution causes increases of the order of 10 to 15% of its best cost. It is one of the best heuristics for the symmetric TSP because of its adaptive nature. Indeed, it swaps a number of partial sequences of the tour. This number is not predetermined; yet, it offers a good tradeoff between solution quality and runtime. While 2-opt and 3-opt break 2 and 3 edges of the tour, LK chooses the number of edges to be broken such that this number yields a minimal cost tour. In this sense, LK may be perceived as a variable-k exchange of k edges. It chooses k links to exchange and tests the utility of exchanging $k+1$ links. (Initially $k = 2$.) Any exchange must generate a feasible neighbor. Its utility is assessed via the difference of the costs of the current solution and its neighbor. It is only adopted when it reduces the current solution's cost. LK marks the exchanged edges yielding the best net cost reduction as permanent and prohibits their elimination for a number of iterations by inserting them into a tabu list. When the exploration of exchanging $k + 1$ links reduces the incumbent's cost, LK updates the incumbent, and reduces k; otherwise, it increases k. LK stops when the incumbent can no longer be improved. Even though the complexity of LK is not well determined in the literature, our implementation has a worst time complexity of $O(m^5)$: It binds k to 5.

5. Computational Results

The computational investigation assesses the performance of hybridization in general, and of its type, in particular, on the solution quality and on the runtime of BS. For this purpose, it uses five versions of BS:

BS$_0$ A standard width-first beam search of beam width ω,

BS$_1$ BS$_0$ augmented with a local search at each node of the tree,

BS$_2$ BS$_1$ with its best solution subject to a 2-opt,

BS$_3$ BS$_1$ with its best solution subject to a 3-opt, and

BS$_4$ BS$_1$ with its best solution subject to the LK heuristic with k up to 5.

It applies these five versions (coded in C and run on an Intel Core i3-4030U, 1.90 GHz, 4GB RAM) to 36 benchmark instances of EGSTSP, all available at http://www.cs.rhul.ac.uk/home/zvero/GTSPLIB/. Let z^{lit} be the best known solution, and $z_\omega^{BS\cdot}, \cdot = 0, 1, 2, 3, 4$ the corresponding BS. solution value, for a beam width $\omega = 1, 2, 3, 4, 5, 10$, obtained within runtime $t_\omega^\cdot$ (expressed in seconds). For this solution, the percent optimality gap $\Delta_\omega^\cdot = 100\% \frac{z_\omega^\cdot - z^{lit}}{z^{lit}}$. Herein, we analyze the results, reported in Appendix A, focusing on the utility of the low- and high-level hybridization of BS. We then conclude with some useful remarks.

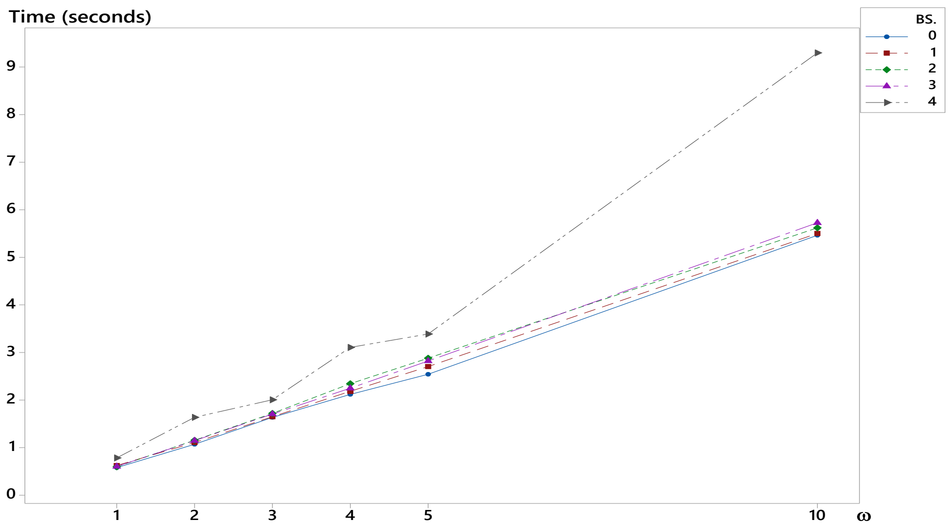

Figure 2: Mean runtime of $\mathrm{BS}_0, \ldots, \mathrm{BS}_4$ as a function of beam width ω

5.1. Utility of the Low Level Hybridization

First, we compare the runtime and solution quality of BS_0 to that of BS_1; that is, of BS without and with local search at each node. (cf. Tables A.1 and A.2 for the detailed results.) We undertake this comparison to highlight the importance of the low-level hybridization undertaken at each node of each level ℓ of the search tree.

Figure 2, which displays the mean runtime of $\mathrm{BS}_0, \ldots, \mathrm{BS}_4$, suggests that the mean runtime of BS increases linearly as a function of the beam width ω. Its average runtime (in seconds) can be estimated as a linear function of ω: $\bar{t}^0 = 0.5454\omega - 0.0459$ and $\bar{t}^1 = 0.5473\omega + 0.0080$, with 99.03% and 99.98% respective coefficients of determination. This behavior is expected as a larger beam width requires more evaluations of partial solutions, of bounds, of sorting, stocking, and retrieving.

Figure 3, which displays box plots of the observed run times, further clarifies this tendency. Yet, it stipulates that the local search does not increase the run time. A statistical paired t-test infers that there is no difference between the mean run times of BS_0 and BS_1 at any level of significance while a paired statistical test infers that the mean Δ_{BS_1} is less than the mean Δ_{BS_0} at any level of significance and that the mean difference $\Delta_{BS_0} - \Delta_{BS_1}$ has a 4.84% point estimate a 4.19% lower bound of a 95% confidence interval. This difference is due to the local search, which enhances the search of BS,

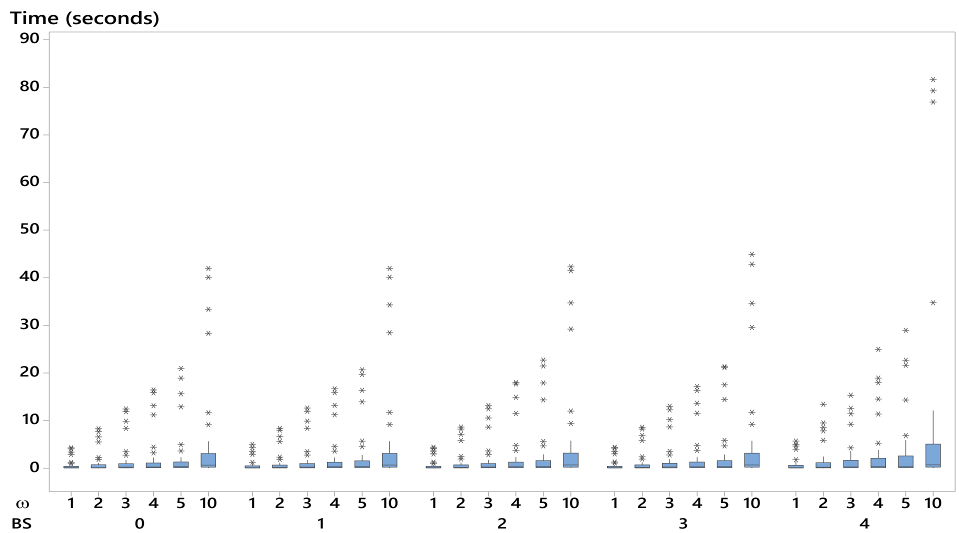

Figure 3: Box plots of observed run times of $\mathrm{BS}_0, \ldots \mathrm{BS}_4$ as a function of beam width ω

by investigating the neighborhood of the partial solution at each node. In fact, $\Delta_{BS_0} > \Delta_{BS_1}$ for all tested instances and for all beam widths. In addition, the average percent deviation $100\% \frac{z^1_\omega - z^0_\omega}{z^0_\omega}$ is of the order of 26%; further highlighting the importance of the local search undertaken by BS_1 at every node. Because BS_1 is superior to BS_0 in terms of solution quality while being equally good in terms of runtime, it can be inferred that BS_1 is better than BS_0.

Figure 4 displays the box plots and means of the percent deviation of the solutions of $\mathrm{BS}_\cdot$, $\cdot = 0, \ldots, 4$, from z^{lit}. Zooming on the box plots and means of BS_0 and BS_1, we detect a seemingly counter-intuitive behavior for small ω. Increasing ω from 1 to 4 does not decrease Δ_{BS_0} and Δ_{BS_1}. This is most likely because it makes BS choose, at a level ℓ, partial solutions that –despite their good quality at level ℓ– do not lead to near-optima. That is, the diversification brought up by the larger beam width focuses on areas of the search space that do not contain the global optimum. The local search undertaken at each node does not mitigate this glitch. On the other hand, increasing ω beyond 5 overcomes this issue. Setting $\omega = 10$ allows BS to obtain solutions that are closer to the global optimum. That is, it makes BS investigate areas of the search space that contain near-global optima. This highlights the importance of the choice of the partial solutions at a level ℓ in

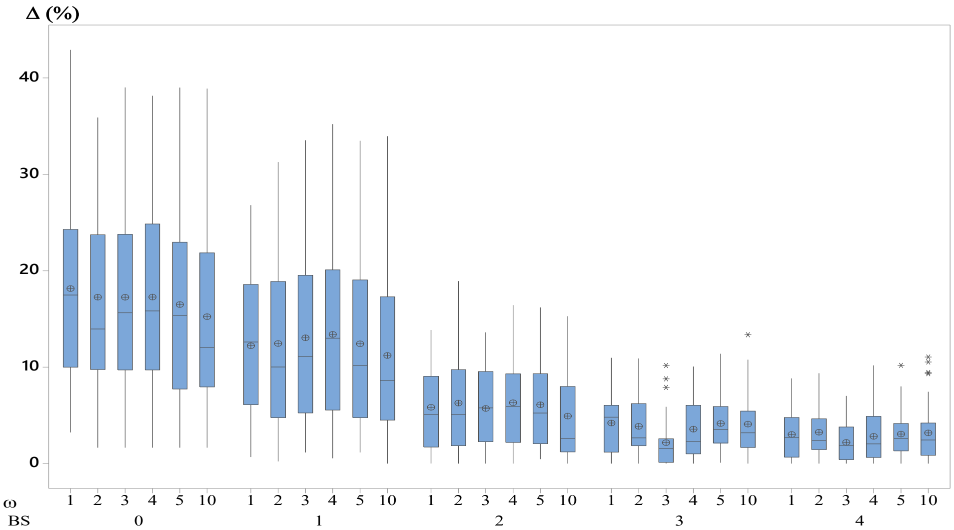

Figure 4: Box plots of percent deviation of the solutions of $BS_0, \ldots, BS_4$ as a function of beam width ω

order to direct the search toward the most promising regions. In this sense, the local search provides a lookahead strategy that helps BS judiciously choose its partial solutions.

5.2. Utility of the High Level Hybridization

Second, we compare the performance of BS_2, BS_3, and BS_4. This comparison highlights the important impact of the high level hybridization, which requires a negligible additional runtime. (cf. Tables A.3 - A.5 for the detailed results.)

As Figure 4 reveals, the improvements of the solution quality due the high-level hybridization are much larger than their counterparts due to the low-level hybridization, regardless of the beam width. These improvements occur at no additional runtime cost except for the last three instances when run with BS_4 and a beam width $\omega = 10$. These instances are marked as outliers in Figure 3, which displays the box plots and means of the observed run times of BS_0 - BS_4. For all beam widths, the mean run time of any of the approaches is larger than its median; signaling the existence of outlier cases, corresponding to the last three instances. Despite the presence of these outliers, which drive the run time of BS_4 up for $\omega = 10$, paired t-tests infer that there is no statistical evidence to claim that the mean run time of

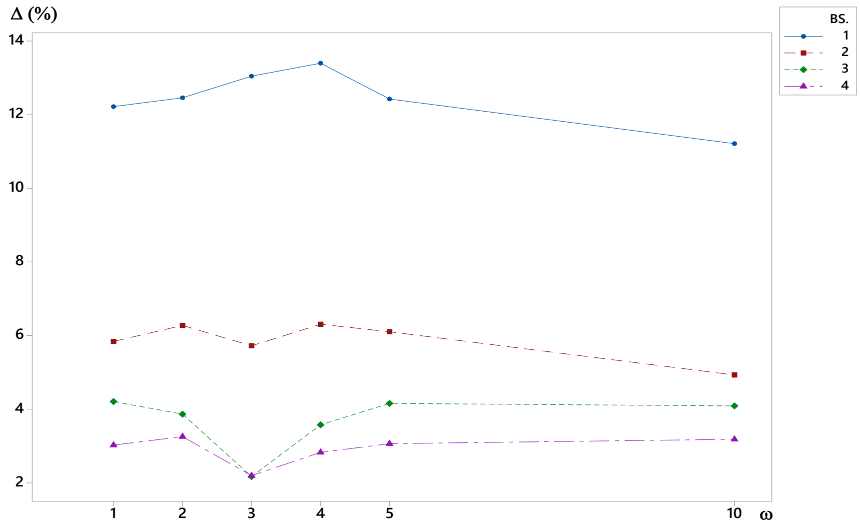

Figure 5: Mean percent deviations of the solutions of $\mathrm{BS}_0, \cdots, \mathrm{BS}_4$ from z^{lit} as a function of beam width ω

any pair of hybridized versions of BS are different at a 5% significance level.

The lack of exploitation and of exploration of the search space makes BS_0 obtain better results for larger beam widthes. This behavior persists for BS_1, which benefits from a local search at each of its nodes, and for BS_2, which benefits from an intensified 2-opt search around its best solution. However, for BS_3 and BS_4, the 3-opt and the LK intensification makes BS obtain its best solutions using a beam width $\omega = 3$, with a mean runtime less than 2 seconds. This is confirmed by Figures 2 and 5, which display respectively the mean percent deviation from z^{lit} and mean runtime as a function of beam width for BS_0 to BS_4.

5.3. Remarks

LK is known to obtain good results when initialized from several random initial solutions. The proposed approach BS_4 provides evidence that it is possible to generate initial solutions for LK in a more systematic manner. Furthermore, the results infer that BS_3 with a beam width $\omega = 3$ yields, on average, better results than the other considered beam searches. However, it remains true that the incumbent of BS_1 can be subjected to three types of searches 2-opt, 3-opt, and LK, at a negligible additional runtime. In fact, there is no statistical difference between the runtime of BS_1 and $\mathrm{BS}_\cdot$, $\cdot = 2, 3, 4$; implying that the bulk of their runtime is caused

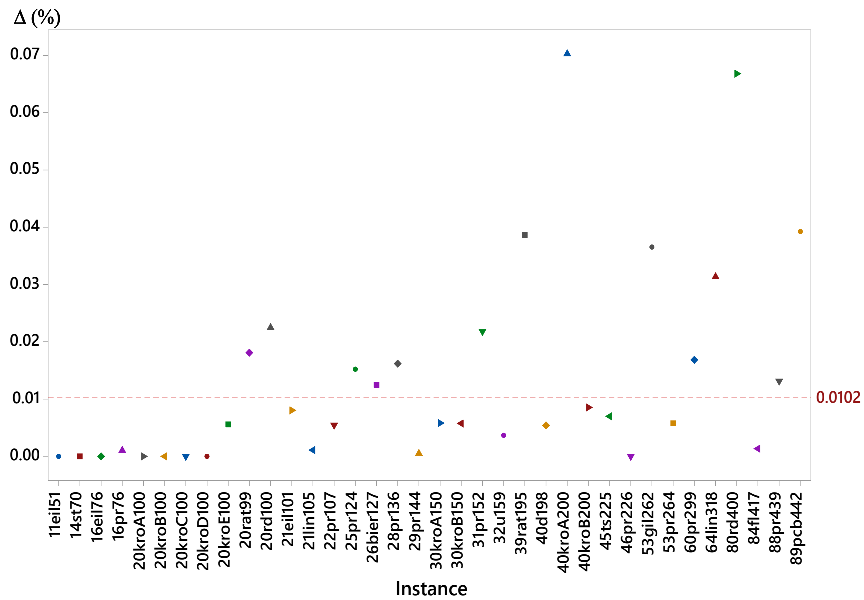

Figure 6: Percent deviation of BS solutions from best known ones

by the BS component. Finally, even though $\omega = 3$ yields in general the best performance, running BS_1 with different beam widths constitutes a good diversification strategy. Using these two additional aforementioned intensification and diversification mechanisms reduces the percent deviation gaps of the BS solution to those observed in the literature; matching the best solution in 22.22% of the instances, and averaging a 0.01344% deviation. The mean should be interpreted with care as it is affected by two outlier values, recorded for instances 40kroA200 and 80rd400, as shown in Figure 6. These outliers are clearly depicted in Figure 7, which displays the resulting box plot of percent deviations for this BS. The corresponding five-point summary of the percent deviation is (Minimum=0, Q1=0.00063, Q2=0.00578, Q3=0.01779, Maximum=0.07027), where Q1, Q2 and Q3 are the first, second and third quartiles. Ignoring the two outlier instances brings the largest deviation over the other 34 instances to 0.03925% and its average to 0.01020%.

6. Conclusion

This paper addressed EGSTSP via a beam search that obtains good solutions for large beam widths. However, to avoid the exponential increase

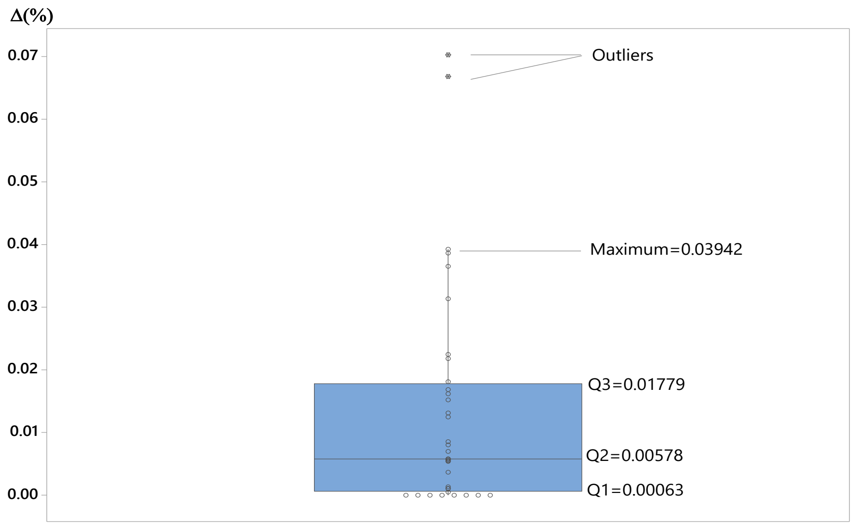

Figure 7: Box plot of percent deviation of BS solutions from best known ones

of runtime associated with branch and bound, we opted for both a low- and a high-level hybridization of the beam search. First, we performed a local search at each node of the tree. This local search acts as a lookahead strategy. It allows the beam search to retain the partial solutions that could lead to near-global optima in lieu of selecting the lowest cost partial solutions. This local search improved the performance of the beam search without affecting its runtime. Second, we subjected the best solution of the beam search to each of three local search operators: 2-Opt, 3-Opt and Lin-Kernighan. This high level hybridization further improved the solution quality of the standard beam search by up to 70% without affecting its runtime. Applying the three search operators to the incumbent offers BS more exploration and exploitation power. The proposed hybridization can be applied to different variants of traveling related problems including vehicle routing, dial-a-ride, and delivery with time windows. Other types of search techniques can also be considered such as simulated annealing, variable neighborhood search, adaptive, and data-driven techniques.

References

[1] Ardalan A., Karimi S., Poursabzi O., Naderi B., 2015, A novel imperialist competitive algorithm for generalized traveling salesman problems. Applied Soft Computing 26:546−555.

[2] Ben-Arieh D., Gutin G., Penn M., Yeo A., Zverovitch A., 2003, Transformations of generalized ATSP into ATSP. Operations Research Letters 31:357−365.

[3] Bontoux B., Artigues C., Feillet D., 2010, A memetic algorithm with a large neighborhood crossover operator for the generalized traveling salesman problem. Computers & Operations Research 37:1844−1852.

[4] Chira C., Pintea C.M., Dumitrescu D., 2007, Sensitive ant systems in combinatorial optimization. Proceedings of the International Conference on Knowledge Engineering, Principles and Techniques, KEPT, Cluj-Napoca (Romania), 185−192.

[5] Dimitrijevic V., Saric Z., 1997, An efficient transformation of the generalized traveling salesman problem into the traveling salesman problem. Digraphs, Information Sciences 102: 105−110.

[6] Fischetti M., Salazar-González J.J., Toth P., 1997, A branch-and-cut algorithm for the symmetric generalized traveling salesman problem. Operations Research 45:378−394.

[7] Fischetti M., Salazar-González JJ., Toth P. (2007) The generalized traveling salesman and orienteering problems. In: Gutin G., Punnen A.P. (eds) The traveling salesman problem and its variations. Combinatorial Optimization, vol 12. Springer, Boston, MA.

[8] Gutin G., Karapetyan D., Krasnogor N., 2008, Memetic algorithm for the generalized asymmetric traveling salesman problem. Studies in Computational Intelligence 129:199−210.

[9] Helsgaun K., 2015, Solving the equality generalized traveling salesman problem using the Lin-Kernighan-Helsgaun-algorithm. Mathematical Programming Computation, 17(3):1−19.

[10] Jiang C., Wan Z., Peng Z., 2020, A new efficient hybrid algorithm for large scale multiple traveling salesman problems. Expert Systems with Applications 139:112867.

[11] Karapetyan D., Gutin G., 2011, Lin-Kernighan heuristic adaptations for the generalized traveling salesman problem. European Journal of Operational Research 208:221−232.

[12] Karapetyan D., Gutin G., 2012, Efficient local search algorithms for known and new neighborhoods for the generalized traveling salesman problem. European Journal of Operational Research 219:234−251.

[13] Khachai M.Y., Neznakhina E.D., 2017, Approximation schemes for the generalized traveling salesman problem. Proceedings of the Steklov Institute of Mathematics 299:97−105.

[14] Laporte G., Mercure H., Nobert Y., 1987, Generalized traveling salesman problem through n sets of nodes: The asymmetric case. Discrete Applied Mathematics 18:185−197.

[15] Lawrence V.S., Daskin M.S., 2006, A random-key genetic algorithm for the generalized traveling salesman problem. European Journal of Operational Research 174:38−53.

[16] Lien Y., Ma E., Wah B.W.S., 1993, Transformation of the generalized traveling salesman problem into the standard traveling salesman problem. Information Sciences 74:177−189.

[17] Lin S., Kernighan B.W., 1973. An effective heuristic Algorithm for the traveling-salesman problem. Operations Research 21:498−516.

[18] Makarovskikh T., Panyukov A., Savitsky E., 2019, Software development for cutting tool routing problems. Procedia Manufacturing 29:567−574.

[19] Mestria M., 2018, New hybrid heuristic algorithm for the clustered traveling salesman problem. Computers & Industrial Engineering 116:1−12.

[20] Noon C., Bean J.C., 1991, A Lagrangian based approach for the asymmetric generalized traveling salesman problem. Operations Research 39(4):623−632.

[21] Petunin A.A., Polishchuk E.G., Ukolov S.S., 2019, On the new algorithm for solving continuous cutting problems. IFAC 52(13):2320−2325

[22] Renaud J., Boctor F.F., Laporte G., 1996, A fast composite heuristic for the symmetric traveling salesman problem. INFORMS Journal on Computing 8(2):134−143.

[23] Renaud J., Boctor F.F., 1998, An efficient composite heuristic for the symmetric generalized travelling salesman problem. European Journal of Operational Research 108(3):571−584.

[24] Salman R., Ekstedt F., Damaschke P., 2020, Branch-and-bound for the precedence constrained generalized traveling salesman problem, Operations Research Letters 48(2):163−166.

[25] Smith S.L., Imeson F., 2017, GLNS: An effective large neighborhood search heuristic for the generalized traveling salesman problem. Computers & Operations Research 87:1−19.

[26] Srivastava S.S., Kumar S., Garg R., Sen P., 1969, Generalized traveling salesman problem through n sets of nodes. CORS Journal 7:97−101.

[27] Sundar K., Rathinam S., 2016, Generalized multiple depot traveling salesmen problem-Polyhedral study and exact algorithm. Computers & Operations Research 70:39−55.

[28] Yang J., Shi X., Marchese M., Liang Y., 2008, An ant colony optimization method for generalized TSP problem. Progress in Natural Science 18:1417−1422.

[29] Yuan Y., Cattaruzza D., Ogier M., Semet F., 2020, A branch-and-cut algorithm for the generalized traveling salesman problem with time windows, European Journal of Operational Research 286(3):849−866.

[30] Zhou X., Rodrigues B., 2017, An extension of the Christofides heuristic for the generalized multiple depot multiple traveling salesmen problem, European Journal of Operational Research 257(3):735−745.

Appendix A. Detailed Computational Results

The results of BS$\cdot$, $\cdot = 0, \ldots, 4$ are reported in Tables A.1-A.5. The first column indicates the '.gts' label of the instance whereas the second column reports its best known solution z^{lit}, available in the literature. The next six triplets of columns report the BS. solution value $z^{BS\cdot}_{\omega}, \cdot = 0, \ldots, 4$, its percent optimality gap $\Delta^{\cdot}_{\omega} = 100\% \frac{z^{\cdot}_{\omega} - z^{lit}}{z^{lit}}$, and its runtime $t^{\cdot}_{\omega}$ in seconds when the beam width $\omega = 1, 2, 3, 4, 5, 10$.

Table A.1: BS_0 Results

		$\omega=1$			$\omega=2$			$\omega=3$			$\omega=4$			$\omega=5$			$\omega=10$		
Instance	z^{lit}	z_1^0	Δ_1^0	t_1^0	z_2^0	Δ_2^0	t_2^0	z_3^0	Δ_3^0	t_3^0	z_4^0	Δ_4^0	t_4^0	z_5^0	Δ_5^0	t_5^0	z_{10}^0	Δ_{10}^0	t_{10}^0
11eil51	174	191	9.77	0.000	189	8.62	0.000	191	9.77	0.000	191	9.77	0.016	190	9.20	0.016	190	9.20	0.016
14st70	316	350	10.76	0.000	346	9.49	0.000	345	9.18	0.016	350	10.76	0.031	341	7.91	0.031	338	6.96	0.062
16eil76	209	260	24.40	0.015	256	22.49	0.015	230	10.05	0.031	222	6.22	0.032	223	6.70	0.047	226	8.13	0.094
16pr76	64925	79114	21.85	0.016	78910	21.54	0.015	78829	21.42	0.031	81513	25.55	0.031	77062	18.69	0.047	71159	9.60	0.078
20kroA100	9711	11169	15.01	0.016	10811	11.33	0.031	11169	15.01	0.047	10856	11.79	0.093	10856	11.79	0.125	10856	11.79	0.203
20kroB100	10328	11136	7.82	0.031	11398	10.36	0.045	12309	19.18	0.047	12511	21.14	0.093	12309	19.18	0.125	11172	8.17	0.203
20kroC100	9554	11841	23.94	0.031	11354	18.84	0.031	12558	31.44	0.062	12639	32.29	0.094	12358	29.35	0.109	11666	22.11	0.203
20kroD100	9450	10380	9.84	0.016	10366	9.69	0.031	10366	9.69	0.062	10366	9.69	0.094	10366	9.69	0.094	10531	11.44	0.203
20kroE100	9523	10469	9.93	0.031	10470	9.94	0.047	10472	9.97	0.062	10676	12.11	0.078	10472	9.97	0.109	10482	10.07	0.203
20rat99	497	610	22.74	0.031	568	14.29	0.047	525	5.63	0.063	527	6.04	0.078	527	6.04	0.110	528	6.24	0.203
20rd100	3650	4201	15.10	0.016	3993	9.40	0.031	3974	8.88	0.062	4119	12.85	0.078	3922	7.45	0.110	3864	5.86	0.203
21eil101	249	293	17.67	0.031	276	10.84	0.047	273	9.64	0.079	273	9.64	0.094	267	7.23	0.125	267	7.23	0.203
21lin105	8213	8828	7.49	0.031	8751	6.55	0.046	8506	3.57	0.078	8468	3.10	0.094	8585	4.53	0.125	8468	3.10	0.209
22pr107	27898	28803	3.24	0.032	28357	1.65	0.062	28357	1.65	0.078	28357	1.65	0.094	28357	1.65	0.140	28357	1.65	0.250
25pr124	36605	39997	9.27	0.031	39413	7.67	0.047	39271	7.28	0.110	40047	9.40	0.187	39076	6.75	0.219	39327	7.44	0.375
26bier127	72418	85261	17.73	0.062	84450	16.61	0.780	84206	16.28	0.156	83981	15.97	0.203	83415	15.19	0.219	80340	10.94	0.391
28pr136	42570	51024	19.86	0.078	48239	13.32	0.094	48239	13.32	0.188	50337	18.25	0.205	50370	18.32	0.297	48740	14.49	0.516
29pr144	45886	47906	4.40	0.094	49868	8.68	0.109	50374	9.78	0.209	50121	9.23	0.265	49411	7.68	0.328	49508	7.89	0.594
30kroA150	11018	13090	18.81	0.109	12728	15.52	0.110	12405	12.59	0.234	12405	12.59	0.250	12728	15.52	0.325	12898	17.06	0.688
30kroB150	12196	13785	13.03	0.110	15940	30.70	0.109	15052	23.42	0.250	15052	23.42	0.250	15052	23.42	0.342	14601	19.72	0.691
31pr152	51576	56853	10.23	0.109	55564	7.73	0.110	53899	4.50	0.250	53899	4.50	0.275	53940	4.58	0.361	53940	4.58	0.703
32u159	22664	27299	20.45	0.125	27428	21.02	0.125	27411	20.95	0.297	27659	22.04	0.312	29099	28.39	0.375	29450	29.94	0.766
39rat195	854	1142	33.72	0.234	1086	27.17	0.250	1066	24.82	0.578	1029	20.49	0.617	1029	20.49	0.812	1028	20.37	1.797
40d198	10557	12337	16.86	0.250	12941	22.58	0.263	12626	19.60	0.609	12497	18.38	0.620	13570	28.54	0.893	13158	24.64	1.906
40kroA200	13406	17224	28.48	0.250	17148	27.91	0.318	17499	30.53	0.625	17020	26.96	0.684	16254	21.24	0.982	17220	28.45	1.984
40kroB200	13111	18736	42.90	0.265	17817	35.89	0.331	17583	34.11	0.625	17805	35.80	0.691	17847	36.12	1.000	17506	33.52	2.000
45ts225	68340	76393	11.78	0.391	77626	13.59	0.518	78478	14.83	0.938	76660	12.17	1.109	76752	12.31	1.212	76504	11.95	3.063
46pr226	64007	69241	8.18	0.375	70401	9.99	0.572	74743	16.77	0.954	74053	15.70	1.061	74047	15.69	1.342	73822	15.33	3.109
53gil262	1013	1349	33.17	0.625	1332	31.49	1.121	1337	31.98	1.703	1310	29.32	2.113	1310	29.32	2.266	1254	23.79	5.610
53pr264	29549	34657	17.29	0.625	33578	13.63	1.113	33769	14.28	1.702	33577	13.63	2.215	33204	12.37	2.283	33321	12.77	5.656
60pr299	22615	29268	29.42	0.984	30592	35.27	1.824	31438	39.01	2.734	31240	38.14	3.215	31433	38.99	3.626	31410	38.89	9.110
64lin318	20765	25908	24.77	1.204	25377	22.21	2.218	25358	22.12	3.453	25250	21.60	4.418	24456	17.78	4.940	24728	19.08	11.640
80rd400	6361	8409	32.20	2.891	8246	29.63	5.524	8057	26.66	8.360	8241	29.56	11.187	8109	27.48	12.891	7919	24.49	28.328
84fl417	9651	10656	10.41	3.391	10959	13.55	6.583	11391	18.03	9.859	11311	17.20	13.131	10951	13.47	15.641	10824	12.15	33.375
88pr439	60099	71001	18.14	4.078	74593	24.12	7.657	74454	23.89	11.860	75318	25.32	15.844	73067	21.58	18.921	72757	21.06	40.094
89pcb442	21657	28768	32.83	4.265	27775	28.25	8.287	28407	31.17	12.437	28014	29.35	16.432	27984	29.21	20.922	27835	28.53	41.953
Summary																			
Minimum			3.24	0.000		1.65	0.000		1.65	0.000		1.65	0.016		1.65	0.016		1.65	0.016
Median			17.48	0.102		13.96	0.110		15.65	0.222		15.84	0.250		15.36	0.327		12.05	0.641
Average			18.15	0.579		17.27	1.070		17.25	1.635		17.27	2.119		16.50	2.542		15.24	5.463
Maximum			42.90	4.265		35.89	8.287		39.01	12.437		38.14	16.432		38.99	20.922		38.89	41.953

Table A.2: BS$_1$ Results

		$\omega=1$			$\omega=2$			$\omega=3$			$\omega=4$			$\omega=5$			$\omega=10$		
Instance	z^{lit}	z_1^1	Δ_1^1	t_1^1	z_2^1	Δ_2^1	t_2^1	z_3^1	Δ_3^1	t_3^1	z_4^1	Δ_4^1	t_4^1	z_5^1	Δ_5^1	t_5^1	z_{10}^1	Δ_{10}^1	t_{10}^1
11eil51	174	181	4.02	0.016	186	6.90	0.000	183	5.17	0.016	183	5.17	0.016	179	2.87	0.016	179	2.87	0.031
14st70	316	326	3.16	0.015	320	1.27	0.015	336	6.33	0.016	320	1.27	0.031	331	4.75	0.031	331	4.75	0.062
16eil76	209	249	19.14	0.016	245	17.22	0.015	218	4.31	0.031	213	1.91	0.031	218	4.31	0.047	213	1.91	0.094
16pr76	64925	76678	18.10	0.016	77347	19.13	0.015	78152	20.37	0.031	78143	20.36	0.031	73175	12.71	0.047	70491	8.57	0.078
20kroA100	9711	10772	10.93	0.016	10234	5.39	0.047	10772	10.93	0.063	10234	5.39	0.094	10234	5.39	0.125	10234	5.39	0.203
20kroB100	10328	10398	0.68	0.032	10352	0.23	0.046	12061	16.78	0.061	12061	16.78	0.094	11332	9.72	0.109	10328	0.00	0.203
20kroC100	9554	11026	15.41	0.031	10629	11.25	0.047	11886	24.41	0.063	12056	26.19	0.094	11585	21.26	0.109	10736	12.37	0.203
20kroD100	9450	10037	6.21	0.016	10022	6.05	0.031	10022	6.05	0.062	10022	6.05	0.094	10022	6.05	0.109	10030	6.14	0.203
20kroE100	9523	9770	2.59	0.031	9771	2.60	0.047	9882	3.77	0.062	9853	3.47	0.093	9882	3.77	0.109	9975	4.75	0.203
20rat99	497	567	14.08	0.032	529	6.44	0.016	519	4.43	0.062	519	4.43	0.079	519	4.43	0.110	519	4.43	0.125
20rd100	3650	4145	13.56	0.016	3926	7.56	0.047	3733	2.27	0.062	4086	11.95	0.078	3824	4.77	0.110	3804	4.22	0.203
21eil101	249	282	13.25	0.031	255	2.41	0.047	264	6.02	0.078	264	6.02	0.094	255	2.41	0.125	255	2.41	0.209
21lin105	8213	8295	1.00	0.032	8357	1.75	0.047	8324	1.35	0.078	8258	0.55	0.109	8354	1.72	0.140	8258	0.55	0.218
22pr107	27898	28222	1.16	0.032	28223	1.16	0.062	28223	1.16	0.078	28223	1.16	0.125	28223	1.16	0.140	28223	1.16	0.250
25pr124	36605	38832	6.08	0.031	38271	4.55	0.093	38623	5.51	0.141	39878	8.94	0.188	38897	6.26	0.219	38535	5.27	0.375
26bier127	72418	84474	16.65	0.063	83663	15.53	0.109	80032	10.51	0.156	83385	15.14	0.187	82492	13.91	0.235	79157	9.31	0.391
28pr136	42570	48468	13.85	0.094	43772	2.82	0.156	43772	2.82	0.188	46484	9.19	0.266	47099	10.64	0.297	46252	8.65	0.516
29pr144	45886	46907	2.23	0.094	48616	5.95	0.172	49431	7.73	0.218	48889	6.54	0.266	48686	6.10	0.344	48783	6.31	0.609
30kroA150	11018	11756	6.70	0.110	12046	9.33	0.188	11825	7.32	0.265	11825	7.32	0.313	12046	9.33	0.375	12081	9.65	0.703
30kroB150	12196	13431	10.13	0.110	15576	27.71	0.188	14548	19.29	0.250	14548	19.29	0.312	14548	19.29	0.390	13718	12.48	0.703
31pr152	51576	53558	3.84	0.125	52835	2.44	0.188	52907	2.58	0.266	52907	2.58	0.344	52830	2.43	0.406	52830	2.43	0.765
32u159	22664	25603	12.97	0.140	26673	17.69	0.219	25948	14.49	0.297	26673	17.69	0.375	28041	23.72	0.438	28109	24.02	0.797
39rat195	854	1048	22.72	0.235	1009	18.15	0.422	1018	19.20	0.578	988	15.69	0.734	988	15.69	0.921	992	16.16	1.812
40d198	10557	12039	14.04	0.250	12696	20.26	0.438	12373	17.20	0.610	12252	16.06	0.781	13206	25.09	0.969	12957	22.73	1.922
40kroA200	13406	16417	22.46	0.281	16367	22.09	0.438	16670	24.35	0.625	16481	22.94	0.828	15591	16.30	1.000	16755	24.98	2.000
40kroB200	13111	16553	26.25	0.331	17172	30.97	0.437	16553	26.25	0.625	17257	31.62	0.828	17406	32.76	1.000	17018	29.80	2.000
45ts225	68340	72930	6.72	0.438	74311	8.74	0.656	75136	9.94	0.953	73622	7.73	1.234	73924	8.17	1.500	73686	7.82	3.078
46pr226	64007	68654	7.26	0.485	69699	8.89	0.656	73558	14.92	0.984	73009	14.06	1.266	73013	14.07	1.562	73050	14.13	3.114
53gil262	1013	1238	22.21	0.687	1291	27.44	1.156	1290	27.34	1.703	1246	23.00	2.234	1238	22.21	2.765	1208	19.25	5.658
53pr264	29549	32711	10.70	0.641	32711	10.70	1.157	32877	11.26	1.703	32637	10.45	2.250	32166	8.86	2.797	31981	8.23	5.671
60pr299	22615	27335	20.87	1.015	29687	31.27	1.843	30197	33.53	2.750	30576	35.20	3.593	30185	33.47	4.500	30293	33.95	9.187
64lin318	20765	24657	18.74	1.219	24267	16.86	2.328	24352	17.27	3.478	24706	18.98	4.578	23931	15.25	5.687	23907	15.13	11.731
80rd400	6361	8066	26.80	2.906	7955	25.06	5.610	7812	22.81	8.375	7986	25.55	11.234	7791	22.48	13.937	7649	20.25	28.438
84fl417	9651	10557	9.39	3.516	10854	12.47	6.625	11310	17.19	9.875	11205	16.10	13.203	10875	12.68	16.359	10749	11.38	34.297
88pr439	60099	67460	12.25	4.306	70902	17.98	7.985	71885	19.61	11.954	73542	22.37	15.875	71110	18.32	19.672	70716	17.67	40.113
89pcb442	21657	26806	23.78	5.016	26474	22.24	8.328	27119	25.22	12.644	27121	25.23	16.672	27061	24.95	20.656	26992	24.63	41.953
Summary																			
Minimum			0.68	0.015		0.23	0.000		1.16	0.016		0.55	0.016		1.16	0.016		0.00	0.031
Median			12.61	0.102		10.02	0.180		11.10	0.234		13.01	0.289		10.18	0.360		8.61	0.656
Average			12.22	0.623		12.46	1.108		13.05	1.650		13.40	2.184		12.43	2.704		11.22	5.503
Maximum			26.80	5.016		31.27	8.328		33.53	12.644		35.20	16.672		33.47	20.656		33.95	41.953

Table A.3: BS_2 Results

		$\omega=1$			$\omega=2$			$\omega=3$			$\omega=4$			$\omega=5$			$\omega=10$		
Instance	z^{lit}	z_1^2	Δ_1^2	t_1^2	z_2^2	Δ_2^2	t_2^2	z_3^2	Δ_3^2	t_3^2	z_4^2	Δ_4^2	t_4^2	z_5^2	Δ_5^2	t_5^2	z_{10}^2	Δ_{10}^2	t_{10}^2
11eil51	174	175	0.57	0.000	175	0.57	0.000	174	0.00	0.015	174	0.00	0.000	178	2.30	0.015	178	2.30	0.031
14st70	316	320	1.27	0.000	320	1.27	0.000	320	1.27	0.016	320	1.27	0.031	320	1.27	0.031	320	1.27	0.063
16eil76	209	223	6.70	0.016	217	3.83	0.015	214	2.39	0.032	209	0.00	0.031	214	2.39	0.047	209	0.00	0.094
16pr76	64925	68205	5.05	0.016	67766	4.38	0.016	64994	0.11	0.031	68310	5.21	0.031	69163	6.53	0.032	65702	1.20	0.094
20kroA100	9711	9712	0.01	0.032	10063	3.62	0.047	9712	0.01	0.078	10063	3.62	0.094	10063	3.62	0.125	10063	3.62	0.203
20kroB100	10328	10398	0.68	0.016	10352	0.23	0.047	10376	0.46	0.062	10542	2.07	0.094	10376	0.46	0.109	10328	0.00	0.219
20kroC100	9554	10189	6.65	0.015	10449	9.37	0.046	10048	5.17	0.063	9554	0.00	0.093	9669	1.20	0.110	9798	2.55	0.204
20kroD100	9450	9962	5.42	0.015	10022	6.05	0.047	10022	6.05	0.078	10022	6.05	0.094	10022	6.05	0.110	10030	6.14	0.203
20kroE100	9523	9770	2.59	0.015	9771	2.60	0.047	9770	2.59	0.079	9706	1.92	0.094	9770	2.59	0.109	9576	0.56	0.219
20rat99	497	517	4.02	0.031	529	6.44	0.047	506	1.81	0.063	506	1.81	0.094	506	1.81	0.109	506	1.81	0.203
20rd100	3650	3825	4.79	0.016	3926	7.56	0.047	3732	2.25	0.063	4047	10.88	0.078	3824	4.77	0.110	3804	4.22	0.203
21eil101	249	282	13.25	0.031	252	1.20	0.047	257	3.21	0.078	262	5.22	0.093	252	1.20	0.125	255	2.41	0.218
21lin105	8213	8295	1.00	0.031	8357	1.75	0.062	8322	1.33	0.078	8258	0.55	0.109	8354	1.72	0.125	8252	0.47	0.219
22pr107	27898	28050	0.54	0.031	28053	0.56	0.063	28053	0.56	0.094	28053	0.56	0.125	28053	0.56	0.140	28053	0.56	0.250
25pr124	36605	38350	4.77	0.047	37462	2.34	0.110	37462	2.34	0.156	37542	2.56	0.187	38011	3.84	0.218	37598	2.71	0.375
26bier127	72418	80458	11.10	0.047	81095	11.98	0.094	79574	9.88	0.157	79295	9.50	0.187	79295	9.50	0.218	75360	4.06	0.406
28pr136	42570	47990	12.73	0.078	43259	1.62	0.156	43772	2.82	0.188	45231	6.25	0.250	45553	7.01	0.282	44778	5.19	0.516
29pr144	45886	45988	0.22	0.094	47525	3.57	0.156	47660	3.87	0.235	47525	3.57	0.281	47714	3.98	0.328	46382	1.08	0.618
30kroA150	11018	11582	5.12	0.109	11402	3.49	0.203	11812	7.21	0.250	11812	7.21	0.312	11402	3.49	0.390	11292	2.49	0.734
30kroB150	12196	12754	4.58	0.109	14505	18.93	0.203	13260	8.72	0.266	13260	8.72	0.328	13260	8.72	0.391	12708	4.20	0.719
31pr152	51576	53350	3.44	0.125	52700	2.18	0.219	52907	2.58	0.281	52907	2.58	0.360	52830	2.43	0.422	52830	2.43	0.765
32u159	22664	23295	2.78	0.141	24356	7.47	0.219	24105	6.36	0.313	24356	7.47	0.375	26093	15.13	0.453	22919	1.13	0.875
39rat195	854	946	10.77	0.250	939	9.95	0.422	914	7.03	0.578	975	14.17	0.750	975	14.17	0.937	978	14.52	1.859
40d198	10557	11519	9.11	0.266	11061	4.77	0.437	11324	7.27	0.625	11068	4.84	0.813	10707	1.42	1.000	10749	1.82	1.922
40kroA200	13406	15263	13.85	0.266	15789	17.78	0.453	15024	12.07	0.657	15458	15.31	0.860	14584	8.79	1.032	15455	15.28	2.047
40kroB200	13111	14294	9.02	0.266	14784	12.76	0.453	14406	9.88	0.656	14808	12.94	0.844	15004	14.44	1.031	14537	10.88	2.063
45ts225	68340	72215	5.67	0.375	72282	5.77	0.672	75136	9.94	0.969	73622	7.73	1.265	73924	8.17	1.563	69120	1.14	3.141
46pr226	64007	64062	0.09	0.375	64007	0.00	0.688	67667	5.72	0.984	67421	5.33	1.281	67667	5.72	1.609	67667	5.72	3.187
53gil262	1013	1127	11.25	0.594	1113	9.87	1.187	1133	11.85	1.765	1090	7.60	2.312	1142	12.73	2.906	1134	11.94	5.781
53pr264	29549	32224	9.05	0.641	32089	8.60	1.204	32256	9.16	1.750	32016	8.35	2.328	32071	8.53	2.875	32044	8.44	5.781
60pr299	22615	23580	4.27	1.000	26095	15.39	1.922	24804	9.68	2.844	25377	12.21	3.734	24182	6.93	4.688	24122	6.66	9.390
64lin318	20765	22511	8.41	1.281	21885	5.39	2.438	22607	8.87	3.625	23061	11.06	4.781	22849	10.04	5.623	23526	13.30	11.985
80rd400	6361	7230	13.66	2.984	7345	15.47	5.813	7143	12.29	8.625	7406	16.43	11.485	7391	16.19	14.360	7301	14.78	29.219
84fl417	9651	9788	1.42	3.609	9699	0.50	7.141	10214	5.83	10.543	10205	5.74	14.907	9843	1.99	17.907	9877	2.34	34.703
88pr439	60099	65050	8.24	4.172	64348	7.07	8.203	68278	13.61	12.469	65158	8.42	17.958	66324	10.36	21.484	65260	8.59	41.485
89pcb442	21657	23431	8.19	4.406	24188	11.69	8.625	24235	11.90	13.156	23827	10.02	17.688	23766	9.74	22.703	24188	11.69	42.281
Summary																			
Minimum			0.01	0.000		0.00	0.000		0.00	0.015		0.00	0.000		0.46	0.015		0.00	0.031
Median			5.09	0.102		5.08	0.180		5.78	0.243		5.90	0.297		5.24	0.359		2.63	0.669
Average			5.84	0.597		6.28	1.154		5.72	1.720		6.31	2.343		6.11	2.881		4.93	5.619
Maximum			13.85	4.406		18.93	8.625		13.61	13.156		16.43	17.958		16.19	22.703		15.28	42.281

Table A.4: BS_3 Results

Instance	z^{lit}	$\omega=1$ z_1^3	Δ_1^3	t_1^3	$\omega=2$ z_2^3	Δ_2^3	t_2^3	$\omega=3$ z_3^3	Δ_3^3	t_3^3	$\omega=4$ z_4^3	Δ_4^3	t_4^3	$\omega=5$ z_5^3	Δ_5^3	t_5^3	$\omega=10$ z_{10}^3	Δ_{10}^3	t_{10}^3
11eil51	174	175	0.57	0.000	175	0.57	0.000	174	0.00	0.015	174	0.00	0.016	178	2.30	0.015	178	2.30	0.031
14st70	316	320	1.27	0.016	320	1.27	0.016	320	1.27	0.016	320	1.27	0.016	320	1.27	0.015	320	1.27	0.063
16eil76	209	223	6.70	0.016	214	2.39	0.016	209	0.00	0.031	209	0.00	0.032	214	2.39	0.047	209	0.00	0.094
16pr76	64925	68205	5.05	0.016	67766	4.38	0.032	64992	0.10	0.032	65526	0.93	0.032	67281	3.63	0.046	68677	5.78	0.094
20kroA100	9711	9712	0.01	0.016	10063	3.62	0.047	9711	0.00	0.062	10063	3.62	0.093	10063	3.62	0.109	10063	3.62	0.218
20kroB100	10328	10398	0.68	0.031	10352	0.23	0.047	10336	0.08	0.063	10542	2.07	0.094	10376	0.46	0.109	10328	0.00	0.204
20kroC100	9554	10189	6.65	0.031	10449	9.37	0.047	9554	0.00	0.078	9554	0.00	0.094	9669	1.20	0.109	9798	2.55	0.219
20kroD100	9450	9962	5.42	0.032	9962	5.42	0.047	9662	2.24	0.078	9962	5.42	0.094	9962	5.42	0.109	9970	5.50	0.203
20kroE100	9523	9770	2.59	0.031	9771	2.60	0.046	9770	2.59	0.062	9706	1.92	0.094	9770	2.59	0.125	9706	1.92	0.203
20rat99	497	517	4.02	0.016	529	6.44	0.047	506	1.81	0.062	506	1.81	0.078	506	1.81	0.109	506	1.81	0.203
20rd100	3650	3825	4.79	0.031	3926	7.56	0.047	3732	2.25	0.063	3816	4.55	0.094	3824	4.77	0.125	3804	4.22	0.203
21eil101	249	271	8.84	0.031	255	2.41	0.063	251	0.80	0.078	262	5.22	0.109	255	2.41	0.125	255	2.41	0.210
21lin105	8213	8295	1.00	0.032	8357	1.75	0.062	8222	0.11	0.078	8258	0.55	0.109	8354	1.72	0.141	8258	0.55	0.250
22pr107	27898	28050	0.54	0.032	28053	0.56	0.062	28053	0.56	0.094	28053	0.56	0.109	28053	0.56	0.156	28053	0.56	0.266
25pr124	36605	38350	4.77	0.047	37481	2.39	0.109	37162	1.52	0.156	37542	2.56	0.172	38011	3.84	0.235	37598	2.71	0.375
26bier127	72418	76799	6.05	0.047	73442	1.41	0.109	73322	1.25	0.172	73402	1.36	0.187	77422	6.91	0.221	74423	2.77	0.390
28pr136	42570	47241	10.97	0.078	43259	1.62	0.156	43259	1.62	0.234	45231	6.25	0.250	45523	6.94	0.297	44778	5.19	0.562
29pr144	45886	45988	0.22	0.094	47525	3.57	0.187	45909	0.05	0.234	47525	3.57	0.281	47716	3.99	0.391	47813	4.20	0.625
30kroA150	11018	11582	5.12	0.110	11402	3.49	0.188	11402	3.49	0.265	11812	7.21	0.283	11402	3.49	0.438	11538	4.72	0.734
30kroB150	12196	12813	5.06	0.110	12530	2.74	0.203	12266	0.57	0.266	12366	1.39	0.343	12550	2.90	0.391	12708	4.20	0.763
31pr152	51576	53350	3.44	0.125	52700	2.18	0.203	52814	2.40	0.281	52907	2.58	0.360	52830	2.43	0.422	52830	2.43	0.781
32u159	22664	22747	0.37	0.141	24356	7.47	0.219	22748	0.37	0.297	24356	7.47	0.375	25247	11.40	0.469	23005	1.50	0.875
39rat195	854	902	5.62	0.250	904	5.85	0.421	902	5.62	0.594	940	10.07	0.750	940	10.07	0.937	944	10.54	1.859
40d198	10557	11128	5.41	0.266	10811	2.41	0.453	10614	0.54	0.625	10614	0.54	0.812	10807	2.37	0.983	10619	0.59	1.937
40kroA200	13406	14672	9.44	0.250	14868	10.91	0.453	14584	8.79	0.688	14685	9.54	0.844	14584	8.79	1.031	14771	10.18	2.031
40kroB200	13111	13804	5.29	0.297	14129	7.76	0.453	13326	1.64	0.688	13326	1.64	0.890	13841	5.57	1.031	13625	3.92	2.031
45ts225	68340	70909	3.76	0.391	69990	2.41	0.672	69501	1.70	0.925	69467	1.65	1.297	69770	2.09	1.563	69531	1.74	3.125
46pr226	64007	64007	0.00	0.375	64007	0.00	0.688	64007	0.00	1.047	64062	0.09	1.297	64062	0.09	1.610	67767	5.87	3.172
53gil262	1013	1076	6.22	0.656	1064	5.03	1.203	1093	7.90	1.875	1097	8.29	2.313	1097	8.29	2.375	1109	9.48	5.719
53pr264	29549	30132	1.97	0.672	30182	2.14	1.218	30166	2.09	1.766	30108	1.89	2.297	30216	2.26	2.875	30840	4.37	5.750
60pr299	22615	23512	3.97	1.078	24234	7.16	1.906	23245	2.79	2.781	23313	3.09	3.735	23608	4.39	4.688	23806	5.27	9.219
64lin318	20765	22207	6.94	1.297	21737	4.68	2.391	21880	5.37	3.531	22741	9.52	4.782	22020	6.04	5.844	22678	9.21	11.750
80rd400	6361	6795	6.82	3.000	6887	8.27	5.828	7008	10.17	8.656	6976	9.67	11.547	7023	10.41	14.438	7211	13.36	29.563
84fl417	9651	9763	1.16	3.515	9673	0.23	6.860	9665	0.15	10.203	9716	0.67	13.641	9730	0.82	17.500	9664	0.13	34.641
88pr439	60099	63001	4.83	4.218	61635	2.56	8.219	61637	2.56	12.266	63791	6.14	16.312	64345	7.07	21.312	61088	1.65	44.922
89pcb442	21657	22966	6.04	4.391	23031	6.34	8.578	22932	5.89	13.000	22901	5.74	17.141	22841	5.47	21.156	23992	10.78	42.828
Summary																			
Average			4.21	0.604		3.87	1.147		2.17	1.705		3.58	2.249		4.16	2.821		4.09	5.725
Minimum			0.00	0.000		0.00	0.000		0.00	0.015		0.00	0.016		0.09	0.015		0.00	0.031
Maximum			10.97	4.391		10.91	8.578		10.17	13.000		10.07	17.141		11.40	21.312		13.36	44.922
Median			4.81	0.102		2.67	0.188		1.57	0.250		2.32	0.282		3.55	0.391		3.20	0.680

Table A.5: BS_4 Results

Instance	z^{lit}	$\omega=1$ z_1^{LK}	Δ_1^{LK}	t_1^{LK}	$\omega=2$ z_2^{LK}	Δ_2^{LK}	t_2^{LK}	$\omega=3$ z_3^{LK}	Δ_3^{LK}	t_3^{LK}	$\omega=4$ z_4^{LK}	Δ_4^{LK}	t_4^{LK}	$\omega=5$ z_5^{LK}	Δ_5^{LK}	t_5^{LK}	$\omega=10$ z_{10}^{LK}	Δ_{10}^{LK}	t_{10}^{LK}
11eil51	174	175	0.57	0.000	175	0.57	0.000	174	0.00	0.031	174	0.00	0.016	179	2.87	0.016	179	2.87	0.031
14st70	316	317	0.32	0.000	317	0.32	0.016	316	0.00	0.031	317	0.32	0.015	316	0.00	0.031	317	0.32	0.062
16eil76	209	211	0.96	0.016	214	2.39	0.031	214	2.39	0.047	209	0.00	0.031	214	2.39	0.063	209	0.00	0.094
16pr76	64925	66783	2.86	0.016	67082	3.32	0.109	64994	0.11	0.047	65646	1.11	0.109	67281	3.63	0.094	65702	1.20	0.109
20kroA100	9711	9712	0.01	0.031	10063	3.62	0.063	9712	0.01	0.125	10063	3.62	0.110	10063	3.62	0.125	10063	3.62	0.218
20kroB100	10328	10328	0.00	0.031	10352	0.23	0.047	10376	0.46	0.078	10542	2.07	0.121	10376	0.46	0.125	10328	0.00	0.203
20kroC100	9554	10189	6.65	0.032	10449	9.37	0.047	9896	3.58	0.078	9554	0.00	0.093	9669	1.20	0.125	9798	2.55	0.234
20kroD100	9450	9462	0.13	0.032	9462	0.13	0.063	9450	0.00	0.109	9462	0.13	0.125	9450	0.00	0.140	9450	0.00	0.234
20kroE100	9523	9770	2.59	0.031	9776	2.66	0.047	9770	2.59	0.078	9706	1.92	0.094	9770	2.59	0.125	9651	1.34	0.218
20rat99	497	517	4.02	0.031	529	6.44	0.047	506	1.81	0.078	506	1.81	0.094	506	1.81	0.125	506	1.81	0.203
20rd100	3650	3825	4.79	0.047	3926	7.56	0.047	3732	2.25	0.078	3873	6.11	0.110	3824	4.77	0.125	3804	4.22	0.234
21eil101	249	271	8.84	0.047	255	2.41	0.078	257	3.21	0.078	262	5.22	0.109	255	2.41	0.125	255	2.41	0.250
21lin105	8213	8295	1.00	0.047	8357	1.75	0.125	8224	0.13	0.094	8252	0.47	0.109	8354	1.72	0.141	8252	0.47	0.234
22pr107	27898	28050	0.54	0.047	28053	0.56	0.106	28053	0.56	0.188	28053	0.56	0.219	28053	0.56	0.219	28053	0.56	0.328
25pr124	36605	38295	4.62	0.063	37462	2.34	0.125	37462	2.34	0.157	37542	2.56	0.281	38011	3.84	0.218	37542	2.56	0.391
26bier127	72418	73442	1.41	0.078	73442	1.41	0.188	73322	1.25	0.156	73884	2.02	0.219	74603	3.02	0.314	74346	2.66	0.422
28pr136	42570	45028	5.77	0.078	43262	1.63	0.250	43262	1.63	0.203	45231	6.25	0.296	45073	5.88	0.437	44778	5.19	0.656
29pr144	45886	45988	0.22	0.094	47525	3.57	0.187	46005	0.26	0.234	47525	3.57	0.406	47402	3.30	0.359	47499	3.52	0.640
30kroA150	11018	11082	0.58	0.125	11202	1.67	0.203	11082	0.58	0.297	11212	1.76	0.328	11202	1.67	0.390	11292	2.49	0.718
30kroB150	12196	12680	3.97	0.141	13001	6.60	0.218	12672	3.90	0.328	12672	3.90	0.359	12672	3.90	0.421	12460	2.16	0.765
31pr152	51576	53350	3.44	0.160	52700	2.18	0.821	52907	2.58	0.375	52907	2.58	0.406	52830	2.43	0.500	52830	2.43	0.844
32u159	22664	23036	1.64	0.219	23017	1.56	0.360	23015	1.55	0.328	23360	3.07	0.392	23442	3.43	0.453	23613	4.19	0.984
39rat195	854	887	3.86	0.297	933	9.25	0.469	887	3.86	0.625	941	10.19	0.750	941	10.19	0.937	944	10.54	1.812
40d198	10557	10771	2.03	0.308	10664	1.01	1.281	10674	1.11	3.359	10674	1.11	3.437	10619	0.59	2.906	10704	1.39	6.750
40kroA200	13406	14372	7.21	0.312	14400	7.41	0.750	14348	7.03	0.813	14360	7.12	1.016	14384	7.30	1.063	14405	7.45	2.188
40kroB200	13111	13722	4.66	0.391	13650	4.11	0.625	13698	4.48	0.672	13223	0.85	0.890	13379	2.04	1.094	13382	2.07	2.079
45ts225	68340	70467	3.11	0.531	69693	1.98	0.687	68817	0.70	0.953	69965	2.38	1.250	70143	2.64	1.547	70029	2.47	3.047
46pr226	64007	64139	0.21	0.645	64007	0.00	0.719	64007	0.00	1.094	64007	0.00	1.312	64347	0.53	1.594	64007	0.00	3.141
53gil262	1013	1067	5.33	0.796	1050	3.65	1.515	1063	4.94	1.844	1081	6.71	2.343	1056	4.24	3.078	1125	11.06	5.719
53pr264	29549	30131	1.97	1.015	30056	1.72	13.422	30135	1.98	4.265	29983	1.47	14.516	29719	0.58	6.828	29769	0.74	7.735
60pr299	22615	22996	1.68	1.422	23705	4.82	2.050	23091	2.10	3.203	23902	5.69	3.813	23598	4.35	4.734	23598	4.35	9.578
64lin318	20765	21842	5.19	1.782	21618	4.11	2.453	22095	6.41	3.609	21416	3.14	5.250	21897	5.45	5.938	22701	9.32	12.125
80rd400	6361	6792	6.78	3.984	6828	7.34	5.844	6786	6.68	9.234	6788	6.71	11.375	6870	8.00	14.328	6962	9.45	34.734
84fl417	9651	9747	0.99	4.651	9675	0.25	7.828	9689	0.39	11.422	9682	0.32	24.953	9714	0.65	22.672	9673	0.23	81.625
88pr439	60099	62942	4.73	5.016	61518	2.36	9.500	62751	4.41	12.594	60888	1.31	18.907	61590	2.48	21.625	61059	1.60	76.891
89pcb442	21657	23020	6.29	5.703	23168	6.98	8.579	22507	3.92	15.312	22905	5.76	17.953	22919	5.83	28.954	23262	7.41	79.250
Summary																			
Average			3.03	0.784		3.26	1.636		2.20	2.006		2.83	3.106		3.07	3.388		3.18	9.299
Minimum			0.00	0.000		0.00	0.000		0.00	0.031		0.00	0.015		0.00	0.016		0.00	0.031
Maximum			8.84	5.703		9.37	13.422		7.03	15.312		10.19	24.953		10.19	28.954		11.06	81.625
Median			2.73	0.110		2.38	0.211		1.90	0.266		2.05	0.344		2.62	0.406		2.45	0.687