# Breakout local search for the traveling salesman problem with job-times

Yuji Zou[a], Jin-Kao Hao[a,*], Qinghua Wu[b]

[a]LERIA, Université d'Angers, 2 bd Lavoisier, 49045 Angers Cedex 01, France
[b]School of Management, Huazhong University of Science and Technology, No. 1037, Luoyu Road, Wuhan, China

## Abstract

The traveling salesman problem with job-times combines two classic NP-hard combinatorial optimization problems: the traveling salesman problem and the scheduling problem. In this problem, a traveler visits sequentially $n$ locations with given travel-times between locations and assigns, to each visited location, one of $n$ jobs with location-dependent job-times. When a job is assigned to a specific location, the job starts to run at that location for its given duration. The goal of the problem is to find a job assignment to minimize the maximum completion time of the $n$ jobs. This work presents an effective heuristic algorithm for the problem based on the breakout local search method. The algorithm combines local search to explore two dedicated neighborhoods and a mixed perturbation to escape local optimum traps. To speed up the search, we introduce a dedicated strategy to identify promising neighboring solutions. We evaluate the algorithm on four sets of 310 benchmark instances in the literature. Computational results show that the proposed algorithm outperforms the previous methods, by reporting improved best results (new upper bounds) for 291 instances and equal best results for 16 other instances. The main search components of the algorithm are investigated to shed light on their contributions to the performance of the algorithm.

*Keywords:* Heuristics; routing-assignment problems; combinatorial optimization; local search.

## 1. Introduction

The Traveling Salesman Problem with Job-times (TSPJ) (Mosayebi et al., 2021) combines the traveling salesman problem (TSP) and the scheduling problem. In this problem, a traveler starts from the depot 0, visits $n$ given locations, and returns to the depot. For each visited location $l$, one job $j$ among $n$ given

*Corresponding author
*Email addresses:* `yujizou6@gmail.com` (Yuji Zou), `jin-kao.hao@univ-angers.fr` (Jin-Kao Hao), `qinghuawu1005@gmail.com` (Qinghua Wu)

jobs with location-dependent processing times (job-times) $jt_{lj}$ is assigned and the job starts to run during that time while the traveler moves to the next location. For a given job $j$ assigned to location $l$, its completion time equals the travel-time from the depot 0 to location $l$ plus the processing time $jt_{lj}$. For a Hamiltonian tour with the $n$ job-location assignment, the completion time of the $n$ jobs is the maximum completion time among the $n$ jobs. The goal of the TSPJ is then to find the Hamiltonian tour starting from and ending at the depot 0 and including the $n$ job-location assignments such that the maximum completion time among the $n$ jobs is minimized. The TSPJ belongs thus to the class of min-max problems and is at least as challenging as its composing problems. A mathematical formulation of the problem presented in (Mosayebi et al., 2021) is provided in Appendix A, which is based on a conventional integer programming formulation for the TSP.

As an illustrative example, Table 1 shows the input data of a TSPJ instance where Nodes denote the depot 0 and the locations. The left and right parts of the table show the travel-times between the nodes and the location-dependent job-times, respectively. Fig. 1 shows two candidate solutions, i.e., two Hamiltonian tours with the job-location assignment together with the completion time of each job. One observes that the solution of Fig. 1(b) with a completion time of 70 is better than the solution of Fig. 1(a) with a completion time of 71.

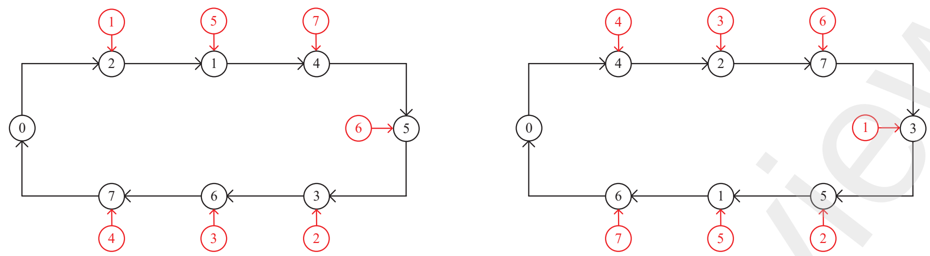
Table 1: An instance of the TSPJ.

| Nodes | Nodes (travel-times) | | | | | | | | Jobs (job-times) | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 0 | 0 | 4 | 6 | 5 | 8 | 9 | 7 | 7 | - | - | - | - | - | - | - | - |
| 1 | 4 | 0 | 8 | 6 | 7 | 5 | 6 | 6 | - | 18 | 20 | 17 | 21 | 19 | 15 | 16 |
| 2 | 6 | 8 | 0 | 5 | 7 | 6 | 10 | 8 | - | 24 | 18 | 19 | 20 | 17 | 26 | 22 |
| 3 | 5 | 6 | 5 | 0 | 3 | 8 | 9 | 5 | - | 20 | 19 | 18 | 21 | 22 | 15 | 24 |
| 4 | 8 | 7 | 7 | 3 | 0 | 6 | 5 | 9 | - | 19 | 17 | 24 | 20 | 21 | 16 | 21 |
| 5 | 9 | 5 | 6 | 8 | 6 | 0 | 8 | 8 | - | 17 | 20 | 19 | 22 | 21 | 23 | 16 |
| 6 | 7 | 6 | 10 | 9 | 5 | 8 | 0 | 5 | - | 17 | 20 | 19 | 22 | 21 | 18 | 23 |
| 7 | 7 | 6 | 8 | 5 | 9 | 8 | 5 | 0 | - | 17 | 20 | 19 | 22 | 21 | 24 | 25 |

It is easy to see that the NP-hard TSP is a special case of the TSPJ when the job-times equal 0. As a result, the TSPJ is at least as difficult as the TSP and solving the problem is computationally challenging.

As discussed in Mosayebi et al. (2021), the TSPJ is a relevant model that can be used to formulate a variety of practical scenarios including autonomous robotics (Bays & Wettergren, 2017), equipment maintenance (Rashidnejad et al., 2018), highly automated manufacturing (Das & Nagendra, 1997), agricultural harvesting (Basnet et al., 2006), and disaster recovery (Barbarosoğlu et al., 2002).

The Sequence-Dependent Robotic Assembly Line Balancing Problem of type 2 (SDRALBP-2) (Lahrichi et al., 2020) is a representative application. In this problem, there are a set of operations, a set of stations and a set of robot types with different abilities. There are three decision problems. One needs to assign the operations to the stations placed in a straight line and sequence the operations in the same station. The precedence relations between the operations need



(a) (b)

Figure 1: Two candidate solutions for the TSPJ instance are shown. The solution of Fig. 1(b) has a completion time of 70, and is thus better than the solution of Fig. 1(a) with a completion time of 71 .

to be satisfied. Finally, one needs to assign a robot to process the operations in each station. The start time of an operation is sequence-dependent since the operation should be processed one by one, and there is a setup time between the operations. The operation processing time is dependent on the robot assigned to the station. The goal of the SDRALBP-2 is to minimize the maximum workload time among all the stations to achieve the balance purpose. Another relevant application is the unrelated parallel machine scheduling problem with sequence and machine-dependent setup times, limited worker resources, and learning effect (Zhang et al., 2021). In this problem, a set of jobs need to be processed with a set of parallel machines. The setup time is sequence and machine-dependent. And the job processing time is dependent on the machine. The purpose of this problem is to minimize the maximum completion time among all jobs. Both applications can be conveniently formulated with the TSPJ model.

Along with the introduction of the TSPJ, Mosayebi et al. (2021) introduced four heuristic algorithms as well as four sets of 310 benchmark instances. They performed comprehensive assessments of these heuristic algorithms on these benchmark instances. They also used the CPLEX MIP solver to solve the integer model shown in Appendix A. We review these heuristic algorithms and related works in Section 2.

Given the relevance of the TSPJ and its computational challenge, it is worth developing effective methods able to provide satisfactory solutions. Currently, such methods are still scarce in the literature, this work aims to fill the gap by presenting a new heuristic algorithm for the TSPJ. Our main contributions are summarized as follows.

We propose an effective algorithm based on the breakout local search method (BLS) (Benlic & Hao, 2013a,b,c) that features two key complementary search components. First, BLS uses a dedicated tabu search procedure to explore candidate solutions based on two specific neighborhoods combined with a neighborhood reduction strategy to make the neighborhood examination more focused.

Second, BLS employes a combined perturbation strategy that adaptively applies a random perturbation and a frequency-based perturbation to help the algorithm to escape local optimum traps.

To show the effectiveness of the proposed algorithm, we carry out extensive computational experiments on the four sets of 310 benchmark instances of (Mosayebi et al., 2021). We show that our BLS algorithm significantly outperforms the existing algorithms in the literature. Specifically, we report improved best-known results (new upper bounds) for 291 instances and equal best-known results for 16 other instances. We present additional experiments to shed light on the influences of the main search components over the performance of the algorithm.

Finally, we will make the code of our algorithm publicly available, which can be used by researchers and practitioners working on the TSPJ and related problems.

In Section 2, we review related works. In Section 3, we present the proposed algorithm. In Section 4, we provide experimental results and comparisons with existing methods. In Section 5, we analyze the main components of the algorithm to shed light on their roles. Conclusions are provided in Section 6.

## 2. Related works

In (Mosayebi et al., 2021), four heuristic algorithms were presented for the TSPJ: TSPJ-NN, reverse assignment, TSPJ-2-*opt*, and local search improvement.

The TSPJ-NN algorithm is adapted from NNH-X, a method extensively used in algorithms for the TSP. NNH-X starts with any node and then chooses the nearest node among the unvisited nodes as the next node to be visited. After all the nodes have been chosen as the first node, the tour with the shortest length is selected as the initial solution. TSPJ-NN extends this method to fit the TSPJ, which assigns the job with the minimum processing time to the location node starting from the last node with every tour produced by the NNH-X method and chooses the solution with the minimum job completion time as the initial solution.

The reverse assignment algorithm is a simple but effective method to assign the jobs to the location nodes. The nodes with a later visiting sequence start the processing later, so assigning the jobs with less processing time to those nodes may shorten the completion time. Based on this idea, the reverse assignment prioritizes the later sequence location nodes and assigns the unselected job with the minimum processing time to the later visited nodes as much as possible. Specifically, job assignment starts from the last node of the tour and moves backward while assigning the unselected jobs with the minimum processing time to every node.

The TSPJ-2-*opt* algorithm is adapted from 2-*opt*, a classical local search heuristic for the TSP (Lin, 1965). TSPJ-2-*opt* executes the job reverse assignment after a 2-*opt* operation. If an improvement is possible, the update is accepted. Then, the algorithm starts from the first node again until no improvement is possible.

The local search improvement algorithm is much more complex. It consists of two steps. In step 1, the job assignment is reconsidered with specific rules. Only when the job assigned to the location node with the maximum completion time is not the job with the minimum processing time in that node, it is possible to improve the solution. If this condition is met, this step will continue. Otherwise, the procedure moves to step 2. In step 1, first, the job with the minimum processing time in the location node with the maximum completion time is assigned to this node, and the other nodes are assigned with the job reverse assignment procedure. If there is any improvement, the change is accepted and one goes to the start point of step 1; otherwise, the procedure continues. Second, exchanging the job on the location node with the maximum completion time with the jobs in later sequence nodes is considered. If there is any improvement, the procedure goes to the start point of step 1; otherwise, goes to step 2. In step 2, the route change is applied by swapping the sequence in the tour of the node with the maximum completion time; first, this node is swapped with its predecessor in the tour; if there is any improvement, the procedure goes to the start point of step 2; otherwise the procedure continues. The second operation is named multi-node swap. Actually, it can be regarded as a 2-*opt* operation that takes the node with the maximum completion time as one of the two points needed in this operation and the other point starting from the first node of the tour. If there is any improvement, the procedure continues and terminates otherwise.

As shown in (Mosayebi et al., 2021), these algorithms have reported competitive results compared to the CPLEX MIP solver. Meanwhile, one notices that these algorithms are of deterministic nature and are all based on the principle of descent search. As a result, they cannot go beyond the local optima they reached at the moment of their termination, even though there would exist local optimal solutions of superior quality. In this work, we investigate a stochastic local search approach (Hoos & Stützle, 2004) that is able to visit multiple local optima solutions to find the best possible solution.

## 3. Breakout local search for the TSPJ

Our proposed algorithm for the TSPJ is based on the general breakout local search method, which has been successively applied to a number of difficult combinatorial optimization problems, such as maximum clique (Benlic & Hao, 2013a), max-cut (Benlic & Hao, 2013b), quadratic assignment (Benlic & Hao, 2013c; Aksan et al., 2017), Steiner tree problem with revenue, budget and hop constraints (Fu & Hao, 2014), assembly sequence planning (Ghandi & Masehian, 2015), and TSP (Krari et al., 2018).

A BLS algorithm iterates a dedicated local search procedure to find high-quality local optimal solutions and an adaptive perturbation procedure to escape local optimum traps. Unlike the conventional iterated local search (Lourenço et al., 2003), which typically applies random perturbations, BLS employs an adaptive multi-perturbation strategy to reach a suitable search diversification. This is achieved by dynamically determining the number of perturbation moves (i.e., the perturbation length) for different types of perturbation (e.g., random or informed perturbations). By iterating the local search phase and the adaptive perturbation phase, BLS favors the balance of search intensification and diversification and helps to better explore the given search space.

In this section, we present the first breakout local search algorithm designed for solving the TSPJ. The BLS algorithm integrates two key complementary ingredients responsible for its effectiveness: a dedicated tabu search procedure exploring two complementary neighborhoods reinforced with a neighborhood reduction technique and a combined perturbation strategy for search diversification.

### *3.1. The BLS procedure*

**Algorithm 1:** Pseudo-code of BLS for the TSPJ

**Input:** Problem instance, time limit $t_{max}$, search depth $\omega$, minimum perturbation length $L_{min}$, maximum perturbation length $L_{max}$.
**Output:** The best solution $S_b$ found so far.

1 $L \leftarrow L_{min}$ ; `/* perturbation length */`
2 $NoImprove \leftarrow 0$ ; `/* counter of consecutive loops` $f_{best}$ `is not improved */`
3 $S_{initial} \leftarrow$ TSPJ-NN() ; `/* generation of initial solution with` TSPJ-NN `*/`
4 $S \leftarrow S_{initial}$ ; `/* current solution */`
5 $S_b \leftarrow S$; `/* best solution found so far */`
6 **while** *$t_{max}$ is not reached* **do**
7     $S_l, S_c \leftarrow$ tabu search$(S, \omega, L_{min})$ ; `/* Section 3.4 */`
8     **if** $f(S_l) < f(S_b)$ **then**
9         $S_b \leftarrow S_l$;
10     $S \leftarrow$ perturbation$(S_c, L)$ ; `/* Section 3.6 */`
11     **if** $L < L_{max}$ **then**
12         $L \leftarrow L + 1$;
13 **return** $S_b$; `/* return the best solution found during the search */`

The proposed BLS algorithm follows the basic framework of the iterated local search. It starts from an initial solution built with the TSPJ-NN heuristic (line 3). Then it alternates iteratively between a tabu search phase and a dedicated perturbation phase (line 7-10). The best solution is updated when the local optimal solution found during the tabu search improves on the best solution during the past search process (line 8-9). Meanwhile, the perturbation length is updated during the tabu search according to the search information. When the tabu search is terminated, the search is considered to stagnate in a local optimum. In this case, the perturbation phase is triggered to modify the current solution $S_c$ with the perturbation length $L$ to help the algorithm to escape the current local optimum. Following this, the perturbation length is increased by 1 so long as it does not reach $L_{max}$. The solution produced by the perturbation becomes the starting point of the next round of the algorithm. The general scheme of BLS is summarized in Algorithm 1.

### 3.2. Search space and solution evaluation

Given a TSPJ instance with $n$ jobs and $n$ locations, a candidate solution can be represented by two $n$-dimensional permutation vectors. Let $\pi_1 : \{1, \ldots, n\} \rightarrow \{1, \ldots, n\}$ be a permutation function representing a TSP tour starting from and ending at the depot 0 and let $\Pi_1$ denote the set of all these permutation functions. Let $\pi_2 : \{1, \ldots, n\} \rightarrow \{1, \ldots, n\}$ be a job-location assignment and let $\Pi_2$ denote the set of all these assignments. Then a candidate solution can be represented as $S = (\pi_1, \pi_2)$, $\pi_1 \in \Pi_1$, $\pi_2 \in \Pi_2$. The search space is formally defined as follows:

$$\Omega = \{(\pi_1, \pi_2) : \pi_1 \in \Pi_1, \pi_2 \in \Pi_2\} \tag{1}$$

Given a solution $S \in \Omega$ in the search space, its quality is assessed by the evaluation function $f$:

$$f(S) = \max(\max(TS_l + \sum_{j=1}^{n} Z_{lj} jt_{lj}), TS_l + Z_{l0} jt_{l0}) \quad \forall l = 1, \ldots, n \tag{2}$$

where $TS_l$ is the processing start time of the job assigned to location node $l$, the binary variable $Z_{lj}$ equals 1 or 0 according to whether job $j$ is assigned to location node $l$ or not, and $jt_{lj}$ is the job processing time of job $j$ in location node $l$.

### 3.3. Initial solution

BLS uses the TSPJ-NN heuristic of (Mosayebi et al., 2021) to obtain an initial solution of reasonable quality. Compared with NNH-X for the TSP, TSPJ-NN considers not only the traveling time but also the job processing time. It can be seen as an extension of NNH-X, which executes the job reverse procedure every time a tour is obtained with a selected node as the first node of the tour. The specific steps of TSPJ-NN are as follows:

- Select any node as the starting node, then choose the nearest unvisited node as the next node; continue this until a TSP tour is obtained. Start from the last node of the tour and assign the unassigned job with the minimum processing time to this node; continue this operation backward till all jobs have been assigned. Calculate the latest job completion time as the objective of the generated solution.

- After all the nodes have been used as the first node of a tour, choose the solution with the minimum completion time as the initial solution.

### 3.4. Examination of candidate solutions with tabu search

BLS uses tabu search (TS) (Glover & Laguna, 1998) to examine candidate solutions of the search space defined in Section 3.2. In this section, we present the general tabu search procedure while the two neighborhoods it explores are presented in Section 3.5.

The general scheme of the tabu search procedure is summarized in Algorithm 2 while its main components are presented in the following subsections. Starting from a given input solution $S$, the algorithm iteratively examines other candidate solutions by exploring two neighborhoods. At each iteration, TS chooses the best eligible neighboring solution among the available neighboring solutions to become the current solution $S_c$ (lines 4-7). Since a neighborhood reduction is applied, it may happen that no neighboring solution is available. In this case, the current solution is slightly perturbed (with perturbation length of 1) (line 8). Each time the current solution $S_c$ becomes better than the recorded local best solution $S_l$ found by the current tabu search run, $S_l$ is updated by $S_c$ (lines 10-11). If $S_c$ is also better than the global best solution $S_b$ from the BLS algorithm, the counter for consecutive non-improvement loops is reset to 0 and the perturbation length $L$ is reset to its minimum $L_{min}$ (lines 12-14). Otherwise, if the current iteration does not update $S_b$, the consecutive non-improvement counter $NoImprove$ is incremented by 1 (line 15). After the move operation, the tabu list is updated according to the information of the move and the current iteration; meanwhile, the frequencies of the edges or jobs related to the executed move are also updated. When the $NoImprove$ counter reaches $\omega$ (a parameter), the search is considered to be trapped in a deep local optimum. The TS procedure terminates and returns the local best solution $S_l$ and its current solution $S_c$. As shown in Algorithm 1 (Section 3.1), $S_l$ will be used by the BLS algorithm to conditionally update the global best solution, while $S_c$ will be used as input of the perturbation procedure (Section 3.6).

### *3.5. Neighborhoods*

Tabu search examines candidate solutions by exploring two neighborhoods induced by two basic move operators (denoted by 2-*opt* and *j*-*swap*).

#### *3.5.1.* 2-*opt neighborhood*

2-*opt* (Lin, 1965) is a well-known operator for the TSP. The 2-*opt* operator basically deletes two edges of the current tour and adds two new edges (see Fig. 2 for an illustrative example). For the TSP, the objective is related to the tour length difference between the two added edges and the two removed edges. The TSPJ is much different since the job start time in the nodes of the reversed edges is changed, so the objective variance after a 2-*opt* move is much more complex to calculate. The simplest way to obtain the objective value after a 2-*opt* move is to calculate, for each location node of the tour, the arriving time, i.e., job processing start time, thus getting the job completion time. However, this is time-consuming.

To accelerate the neighborhood evaluation, our BLS algorithm uses a neighborhood reduction method to eliminate unpromising neighboring solutions. Let $l_m$ denote the node that has the maximum completion time. As shown in Fig. 2(c), when $l_m$ is visited behind the second node involved in the 2-*opt* move, if $\Delta = D_{ac} + D_{bd} - D_{ab} - D_{cd} > 0$, where $D$ is the distance between any two nodes, then we know that such a move results in a longer tour, increases the

**Algorithm 2:** Pseudo-code of tabu search

**Input:** Input solution $S$, reduced neighborhood $N_1$, $N_2$, search depth $\omega$, minimum perturbation length $L_{min}$, current global best solution found so far $S_b$.
**Output:** The current solution $S_c$, the local optimal solution found during tabu search $S_l$.

$NoImprove \leftarrow 0$ ; `/* initialization of non-improvement iteration counter */`
$S_c \leftarrow S$; `/*` $S_c$ `is the current solution */`
$S_l \leftarrow S$; `/*` $S_l$ `records the local best solution found during tabu search */`
**while** $NoImprove < \omega$ **do**
    **if** $N_1(S_c) \cup N_2(S_c) \neq \emptyset$ **then**
        Choose the best eligible neighboring solution $S' \in N_1(S_c) \cup N_2(S_c)$;
        $S_c \leftarrow S'$;
    **else**
        $S_c \leftarrow perturbation(S_c, 1)$;
    **if** $f(S_c) < f(S_l)$ **then**
        $S_l \leftarrow S_c$;
    **if** $f(S_c) < f(S_b)$ **then**
        $NoImprove \leftarrow 0$;
        $L \leftarrow L_{min}$;
    **else**
        $NoImprove \leftarrow NoImprove + 1$;
    Update the tabu list $TL$ and the frequency matrices $F_e$,$F_j$;
**return** $S_l$, $S_c$;

job start time in $l_m$, and leads to the increase of the finishing time. Thus, it is no use to consider such moves in the search process. In our BLS, we ignore them. We have observed that $l_m$ is often visited in the later sequence. So most of the solutions produced by a 2-*opt* move are as shown in Fig. 2(c), that are not considered by BLS.

Formally, the reduced neighborhood $N_1$ is defined as Eqs. (3) - (5), where $S$ is the given solution, $N'_1(S)$ is the whole 2-*opt* neighborhood, and $N_{f1}(S)$ is the set of filtered neighborhood, $e_1 = (l_a, l_b)$ and $e_2 = (l_c, l_d)$ are the two deleted edges with their nodes $l_a, l_b, l_c, l_d$, $E$ is the edge set, $S \oplus 2\text{-}opt(l_a, l_b, l_c, l_d)$ is the resulting neighboring solution. Let $p$ be the variation representing the position of the location node in the tour and $l_a$ be the $p_a$th location node to be visited. We use $\Delta''$ to denote the tour length variation resulting from the 2-*opt*.

$$N_1(S) = N'_1(S) \setminus N_{f1}(S) \tag{3}$$

$$N'_1(S) = \{S' : S' = S \oplus 2\text{-}opt(l_a, l_b, l_c, l_d), (l_a, l_b) = e_1, (l_c, l_d) = e_2, e_1, e_2 \in E\} \tag{4}$$

$$N_{f1}(S) = \{S'' : S'' = S \oplus 2\text{-}opt(l_a, l_b, l_c, l_d), p_m \geq max\{p_a, p_b, p_c, p_d\}, \Delta'' > 0\} \tag{5}$$

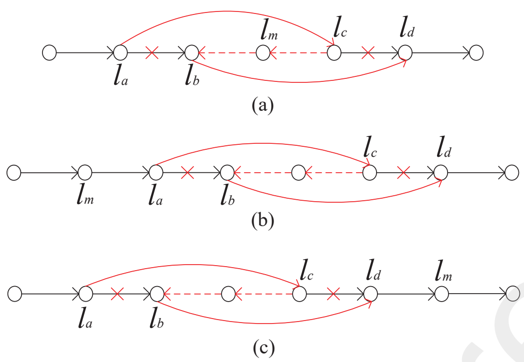



Figure 2: Three possible 2-*opt* moves according to the position of the node with the maximum completion time denoted by $l_m$. (a) $l_m$ belongs to the reversed edge, (b) $l_m$ is before the first node of 2-*opt*, (c) $l_m$ is behind the second node of 2-*opt*.

At each iteration, instead of examining all neighboring solutions in $N_1'(S)$ induced by 2-*opt* move, our BLS algorithm examines the reduced neighborhood $N_{f1}(S)$ to focus on promising solutions only. The size of the $N_1'(S)$ neighborhood is $n*(n-1)/2$. The time complexity of calculating the objective value of a neighboring solution is $O(n)$. So the time complexity of examining $N_1'(S)$ is $O(n^3)$. For the reduced neighborhood $N_{f1}(S)$, there are three cases for the 2-*opt* move, as shown in Fig. 2. For the cases of Fig. 2(a) and 2(b), $O(n)$ time is needed to identify the objective value of a neighboring solution. For the case of Fig. 2(c), the worst situation is that the $\Delta''$ values resulting from the move are all less than 0. Then the time complexity is the same as for $N_1'(S)$, i.e., $O(n^3)$. The best situation is that the $\Delta''$ values are all greater than 0 and $m$ is equal to $n$, the time complexity becomes $O(n^2)$. From the experimental results in Section 5.1, we can observe that the neighborhood reduction can significantly improve the effectiveness of the BLS algorithm.

*3.5.2. j-swap neighborhood*

The *j-swap* operator is inspired by the popular *swap* operator, which is extremely effective for permutation-based assignment problems like quadratic assignment (Benlic & Hao, 2013c, 2015; Taillard, 1991). Since the TSPJ includes a job assignment task, we naturally adopt *j-swap* for this problem.

The *j-swap* operator exchanges the assignments of two jobs (see Fig. 3 for an example). Thus only the job processing times of the two involved nodes are exchanged without changing the processing start time of each exchanged job. It is easy to observe that only changing the job assigned to the node with the maximum completion time (i.e. $l_m$) may decrease (improve) the objective value. So in our algorithm, we constrain *j-swap* to focus on the moves related to that job only. In other words, *j-swap* only changes the job assigned to the location

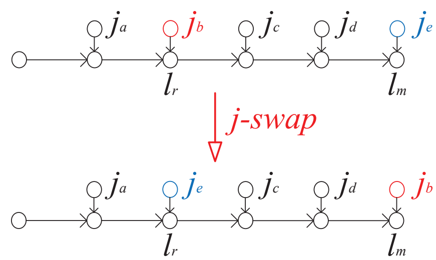



Figure 3: $j$-*swap*, exchange the job $j_e$ assigned to the node having the maximum completion time $l_m$ with another job $j_b$ assigned to location node $l_r$.

with the maximum completion time with another job.

Formally, the reduced $j$-*swap* neighborhood $N_2$ is defined as follows, where $j_b$ and $j_e$ are the jobs exchanged in $j$-*swap*, and $m$ is the index of the location node with the maximum completion time $l_m$.

$$N_2(S) = \{S' : S' = S \oplus j\text{-}swap(j_b, j_e), jt_{mb} < jt_{me}\} \tag{6}$$

As shown in Fig. 3, we try to exchange the job $j_e$ assigned to $l_m$ with another job. Let $jt_{me}$ be the processing time of $j_e$ in the location node $l_m$, let $jt_{mb}$ be the processing time of the job to be changed to $l_m$. Then only when $jt_{mb} < jt_{me}$ is satisfied, we consider this move operation. Clearly the time complexity of our constrained $j$-*swap* operator is $O(n)$, reducing significantly the time complexity $O(n^2)$ for the unconstrained $j$-*swap* operator.

#### *3.5.3. Tabu list management*

Our BLS algorithm employs two tabu lists to avoid short-term cycling: an edge tabu list for 2-*opt* move and a job tabu list for $j$-*swap*. For 2-*opt*, once an edge $e$ is deleted from the solution by the 2-*opt* move, the edge is added to the edge tabu list and is forbidden to be added again to the solution during the next consecutive $\beta$ iterations ($\beta$ is the so-called tabu tenure). So if any of the two new edges used by the 2-*opt* move is in tabu statue, this move will not be performed. Similarly, for $j$-*swap*, when a job $j$ is removed from a location node $l$, the job is forbidden to be assigned to this location node $l$ again during the next consecutive $\beta$ iterations.

During the search process, the best neighboring solution not forbidden by any tabu list is selected to replace the current solution. Notice that the tabu statue of a move is ignored if it can produce a solution better than the best solution ever found, which is called aspiration criterion (Glover & Laguna, 1998).

### *3.6. Combined perturbation*

The tabu mechanism can prevent the search from the short-term cycles, but it may fail to prevent the algorithm from being trapped into deep local optima.

**Algorithm 3:** Pseudo-code of combined perturbation

```
   Input: Input solution S_c, frequency matrices F_e,F_j.
   Output: Perturbed S.
1  Identify the job assigned to the node with the maximum completion time j_m in S_c
    if rand(0,1) < 0.5 then
2      //With probability 0.5, apply random perturbation;
3      if rand(0,1) < 0.5 then
4          e_1, e_2 ← Two randomly choosed edges in S_c;
5          S ← Execute the 2-opt operation with e_1 and e_2;
6      else
7          j_r ← A randomly choosed job;
8          S ← Exchange j_m and j_r;
9  else
10     //With probability 0.5, apply frequency-based perturbation;
11     if rand(0,1) < 0.5 then
12         e_3, e_4 ← Two edges with the least and the second least move frequency;
13         S ← Execute the 2-opt operation with e_3 and e_4;
14     else
15         j_f ← The job with the least move frequency;
16         S ← Exchange j_m and j_f;
17 Update the frequency matrices F_e and F_j;
18 return S;
```

To boost the global diversification of the algorithm, we introduce a dedicated perturbation strategy, which is triggered when the search is judged to be stagnating. According to the general BLS approach, our BLS algorithm for the TSPJ mixes two types of perturbations: informed perturbation (guided by historical search information related to move frequencies) and random perturbation. Indeed, frequency-guided perturbation has proved to be quite successful in several algorithms (Zhou & Hao, 2017; Li et al., 2020), while random perturbation is very popular in iterated local search algorithms.

The perturbation strategy is described in Algorithm 3. We choose the perturbation type with an equal probability (line 1). There are two move operators for perturbation, which are applied with an equal probability as well. For the random perturbation, we randomly select two edges to be replaced for the 2-*opt* move and a job to exchange with the job assigned to the location node with the maximum completion time (lines 3-8). For the frequency-based perturbation, we use the move frequency of the edges and the jobs to guide the perturbation (lines 10-16). For this, we maintain a long-term memory represented by two vectors $F_e$ and $F_j$ to record the number of times an edge or a job is involved in a 2-*opt* or *j-swap* move. We select the edges or location nodes that have the least and second least frequency to perform the perturbation. The specific steps are as follows:

- Initially, set the frequency of all edges and jobs to 0, i.e., $F_e(e) = 0, F_j(j) = 0$ for each edge $e \in E$, job $j \in J$, where $E$ is the edge set and $J$ is the job set.

- Subsequently, during the search process, $F_e$ and $F_j$ are updated each time a

2-*opt* or *j-swap* move is performed.

- Finally, the random or frequency-based perturbation is applied with equal probability and each chosen perturbation performs, with equal probability, either the 2-*opt* or *j-swap* move $L$ times ($L$ is the perturbation length). During the perturbation process, the frequency vectors are updated.

## 4. Computational results

We now report extensive computational results of the proposed BLS algorithm on benchmark instances and comparisons with state-of-the-art algorithms.

### *4.1. Benchmark instances*

Our experiments are based on four sets of 310 instances with different sizes introduced in Mosayebi et al. (2021). The traveling distance between different locations of the instances in Set I is from the TSPLIB (Reinelt, 1991), whose optimal solutions for the TSP are known. The job processing time in different location nodes is generated randomly and is required to be between 50 to 80 percent of the optimal tour length. The traveling time and the job processing time in other sets are all produced randomly. The job processing time is required to be under 50 to 80 percent of the tour length obtained by the NNH-X heuristic and 2-*opt*. More information about these instances can be found in Mosayebi et al. (2021)[1].

**Set I** (10 instances): These instances are constructed based on 10 TSP instances from the TSPLIB: gr17, gr21, gr24, fri26, bays29, gr48, eil51, berlin52, eil76, and eil101.

**Set II, Set III, Set IV** (100 instances per set): The instances from these sets are constructed randomly. The node and job numbers are from 40 to 50, 400 to 500, and 1000 to 1200, respectively, which cover small, medium, and large instances.

### *4.2. Experimental protocol and reference algorithms*

Table 2: Parameters tuning results.

| Parameters | Section | Description | Considered values | Final value |
|---|---|---|---|---|
| $\omega$ | 3.4 | search depth | $\{0.14, 0.56, 0.3, 0.08\}$ | 0.08 |
| $\beta$ | 3.5.3 | tabu tenure | $\{0.35, 0.5, 0.07, 0.14\}$ | 0.07 |
| $L_{min}$ | 3.6 | minimum perturbation length | $\{0.06, 0.03, 0.04, 0.07\}$ | 0.04 |
| $L_{max}$ | 3.6 | maximum perturbation length | $\{0.27, 0.2, 0.15, 0.3\}$ | 0.15 |

**Parameter setting**. BLS has four parameters: tabu tenure $\beta$, search depth $\omega$, minimum perturbation length $l_0$, maximum perturbation length $l_{max}$. In order to calibrate these parameters, we used the "IRACE" (López-Ibáñez et al.,

[1]The instances are available at https://github.com/TSPJLIB

2016) package to automatically identify a set of suitable parameter values. In this experiment, we randomly selected 1 instance from Set I, 3 instances from Set II, Set III, Set IV, respectively. The maximum number of runs (tuning budget) was set to be 1000. The candidate values of these parameters and the final selected values are shown in Table 2.

**Reference algorithms**. For our comparative study, we use as our reference methods the four heuristic algorithms proposed in Mosayebi et al. (2021) (denotated by Pro.I, Pro.II, Pro.III and Pro.IV), which represent the state-of-the-art for solving the TSPJ. Given that the source codes of these algorithms are unavailable, we faithfully re-implemented them, and verified that the results from our implementation match the results initially reported in Mosayebi et al. (2021). In addition to these main reference algorithms, we also run the CPLEX solver on the mathematical model presented in Appendix A with a time limit of 7200 seconds.

**Experimental setting**. BLS and the re-implemented reference algorithms were programmed in C++ and complied with the g++ compiler with the -O3 option. All the experiments were conducted on a computer with an Intel Xeon E5-2670 processor of 2.5 GHz CPU and 6 GB RAM running Linux. In order to eliminate stochastic factors, each algorithm was run 10 times on each instance with a different random seed per run.

**Stopping condition**. The reference algorithms are of deterministic nature and stop when no improvement can be reached. To make a fair comparison between BLS and the reference algorithms, we identify the average running time required by the four reference algorithms from Mosayebi et al. (2021) to reach their best solutions for the 10 instances of Set I and the average running time required by them to find their best solutions for the 100 instances of Set II, Set III and Set IV.

Specifically, for Set I, the cutoff time is set to be 60 seconds for eil101-J, 30 seconds for gr48-J, eil51-J, berlin52-J and eil76-J, 10 seconds for the 5 remaining instances. For the other sets, the cutoff time is 0.0012 seconds for Set II, 2.19 seconds for Set III and 33.93 seconds for Set IV. As such, our BLS algorithm is run under a fair stopping condition compared to the reference algorithms.

Finally, in order to better show the long term behavior of our BLS algorithm, we additionally run BLS with a relaxed stopping condition, 30 seconds for Set II, 50 seconds for Set III and 70 seconds for Set IV.

### *4.3. Computational results and comparison*

This section reports the comparative results between the proposed BLS algorithm and reference algorithms (denotated by Pro.I, Pro.II, Pro.III and Pro.IV). The results are obtained according to the experimental protocol above.

The comparative results of the BLS and the four reference algorithms are summarized in Table 3 while the detailed results on each instance are provided in Appendix B (Table B.6 - Table B.12). In Table 3, the first column indicates the benchmark set. Column 2 presents the cut-off time running by our BLS, for Set I which has detailed results in Mosayebi et al. (2021). Column 3 shows

Table 3: Summary of the number of instances where BLS reports a better (W), equal (T) or worse (L) $f_{best}$ value compared to the results in Mosayebi et al. (2021) including the $p$-values from the Wilcoxon singed-rank test on the benchmark sets between BLS and each reference algorithm Pro.I, Pro.II, Pro.III and Pro.IV.

| Instance | Cut-off time(s) | Pair algorithms | W | T | L | $p$-value |
|---|---|---|---|---|---|---|
| | | BLS vs. BKS | 9 | 1 | 0 | 0.0077 |
| | | BLS vs. Pro.I | 9 | 1 | 0 | 0.0077 |
| Set I | - | BLS vs. Pro.II | 10 | 0 | 0 | 0.0020 |
| | | BLS vs. Pro.III | 9 | 1 | 0 | 0.0077 |
| | | BLS vs. Pro.IV | 9 | 1 | 0 | 0.0077 |
| | | BLS vs. BKS | 46 | 18 | 36 | 0.6662 |
| | | BLS vs. Pro.I | 80 | 13 | 7 | 2.389e-12 |
| Set II | 0.0012 | BLS vs. Pro.II | 69 | 7 | 29 | 1.384e-4 |
| | | BLS vs. Pro.III | 76 | 16 | 8 | 8.332e-12 |
| | | BLS vs. Pro.IV | 79 | 11 | 10 | 3.58e-12 |
| | | BLS vs. BKS | 98 | 0 | 2 | 4.5e-18 |
| | | BLS vs. Pro.I | 100 | 0 | 0 | 3.876e-18 |
| Set III | 2.19 | BLS vs. Pro.II | 98 | 0 | 2 | 4.371e-17 |
| | | BLS vs. Pro.III | 100 | 0 | 0 | 3.877e-18 |
| | | BLS vs. Pro.IV | 100 | 0 | 0 | 3.877e-18 |
| | | BLS vs. BKS | 97 | 0 | 3 | 9.246e-17 |
| | | BLS vs. Pro.I | 99 | 0 | 1 | 4.874e-17 |
| Set IV | 33.93 | BLS vs. Pro.II | 98 | 0 | 2 | 8.479e-16 |
| | | BLS vs. Pro.III | 99 | 0 | 1 | 5.166e-17 |
| | | BLS vs. Pro.IV | 98 | 0 | 2 | 5.399e-17 |
| | | BLS vs. BKS | 84 | 15 | 1 | 1.363e-15 |
| | | BLS vs. Pro.I | 91 | 8 | 1 | 8.58e-17 |
| Set II | 30 | BLS vs. Pro.II | 97 | 3 | 0 | 1.13e-17 |
| | | BLS vs. Pro.III | 88 | 11 | 1 | 2.754e-16 |
| | | BLS vs. Pro.IV | 91 | 8 | 1 | 1.231e-16 |
| | | BLS vs. BKS | 99 | 0 | 1 | 3.98e-18 |
| | | BLS vs. Pro.I | 100 | 0 | 0 | 3.881e-18 |
| Set III | 50 | BLS vs. Pro.II | 99 | 0 | 1 | 3.998e-18 |
| | | BLS vs. Pro.III | 100 | 0 | 0 | 3.882e-18 |
| | | BLS vs. Pro.IV | 100 | 0 | 0 | 3.883e-18 |
| | | BLS vs. BKS | 99 | 0 | 1 | 7.547e-17 |
| | | BLS vs. Pro.I | 99 | 0 | 1 | 3.186e-19 |
| Set IV | 70 | BLS vs. Pro.II | 99 | 0 | 1 | 7.548e-17 |
| | | BLS vs. Pro.III | 99 | 0 | 1 | 3.234e-17 |
| | | BLS vs. Pro.IV | 99 | 0 | 1 | 2.711e-17 |

the compared algorithms including the best-known solutions (BKS). Columns 4 - 6 indicate the number of instances for which BLS obtains a better, equal, or worse $f_{best}$ value compared to each reference algorithm. To check the statistical significance of the compared results, the $p$-values from the Wilcoxon singned-rank test on $f_{best}$ values over the instances from the same set between BLS and the compared algorithms are shown in column 7 and a $p$-value smaller than 0.05 indicates a statistical significant difference.

From the summarized results of Table 3 and detailed results of Appendix B, we observe that our BLS algorithm performs extremely well compared to the algorithms proposed in Mosayebi et al. (2021). In particular, BLS discovers 291 record-breaking results (new upper bounds) out of the 310 instances while matching the best-known results for 16 other instances. BLS reports a slightly worse result only on 3 instances (instance 36 in Set II, instance 48 in Set III, instance 19 in Set IV), with a small gap to the best-known result of 0.38%, 0.03%, and 0.2% respectively.

The $p$-values of Table 3 from the Wilcoxon signed-rank test are much smaller than 0.05 except for the results of Set II with the extreme short running time. Nevertheless, the results of our BLS are still better for most instances. It can be confirmed that the results of our BLS are significantly better than the compared results.

### 4.4. Convergence analysis

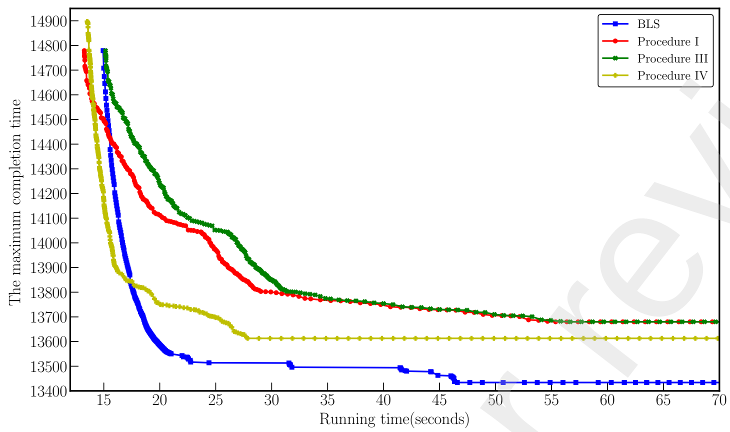



Figure 4: Convergence curves (running profiles) of BLS and three reference procedures for solving instance 61 from set IV.

To illustrate the running behavior of the compared algorithms during the search process, we provide in Fig. 4 their convergence curves (also called running profiles) of the BLS algorithm and three reference procedures on instance 61 from set IV, where the X-axis and Y-axis show the running time in seconds and the best objective value, respectively. The procedure II is ignored in this study because it focuses on the tour length optimization first, then constructs the final solution with the job reverse procedure. Hence, it is impossible to get the objective value of this procedure during the search process.

From Fig. 4, we observe that BLS and Pro.IV improve their best solutions more drastically than Pro.I and Pro.II. More importantly, BLS is able to continue its improvement along the time while the reference procedures fail to do so from some time point. This indicates that BLS has a lasting search capacity, making it possible for the algorithm to reach high-quality solutions that the reference algorithms cannot attain.

## 5. Assessment of algorithmic components

In this section, we analyze the two essential components of our BLS algorithm: neighborhood reduction and frequency-based perturbation.

### 5.1. Importance of the neighborhood reduction

To verify the importance of the neighborhood reduction on the BLS algorithm, we created a BLS variant named BLS-NoReduction, which did not use the neighborhood reduction. We run the two algorithms independently 10 times on the large size instances of set IV, using the parameters in Section 4.2 and a cut-off time of 70 seconds per run and per instance. Table 4 summarizes the comparative results of BLS and BLS-NoReduction on these 100 instances. From these two tables, one observes that BLS significantly dominates BLS-NoReduction, indicating that the performance of BLS deteriorates greatly if the neighborhood reduction is removed from the algorithm. This experiment confirms thus the usefulness of the neighborhood reduction as a critical technique contributing to the effectiveness of the BLS algorithm.

Table 4: Summarized results of BLS and BLS-NoReduction on the 100 large instances of Set III in terms of average result and running time together with the $p$-values from the Wilcoxon singed-rank test.

| | BLS-NoReduction | BLS |
|---|---|---|
| Average | 14932.4 | 13653 |
| $T_{avg}$ | 70.009 | 31.09 |
| $p$-value | 4.96e-18 | |

*5.2. Influence of the frequency-based perturbation*

The frequency-based perturbation is designed to help the algorithm to escape local optimum traps. To show the influence of this strategy, we created a variant of BLS named BLS-Random which only applies the random perturbation mentioned in Section 3.6. We ran BLS-Random on all the instances under the relaxed stopping condition discussed in Section 4.2. The results were summarized in Table 5. In this table, columns 2 and 3 show for BLS and BLS-Random the grand average of the best objective values of all the instances of each set. Columns 5-7 present the number of instances for which the BLS algorithm reached a better, the same and a worse result compared to BLS-Random.

Table 5 shows that BLS with the frequency-based perturbation performs significantly better than the variant with the random perturbation only. This experiment confirms the benefit of the frequency-based perturbation for the performance of the BLS algorithm.

Table 5: Summarized results of BLS and BLS-Random, including the number of instances for which BLS reports a better (W), equal (T) or worse (L) average value compared to BLS-Random and the $p$-values from the Wilcoxon singed-rank test.

| Instance | $\text{BLS}_{Avg}$ | $\text{BLS-Random}_{Avg}$ | $p$-value | W | T | L |
|---|---|---|---|---|---|---|
| Set I | 3729.90 | 3730.41 | - | 2 | 8 | 0 |
| Set II | 280.85 | 281.58 | 0.00018 | 46 | 56 | 8 |
| Set III | 3463.19 | 3469.73 | 3.176e-7 | 62 | 24 | 14 |
| Set IV | 13625.51 | 13642.81 | 4.96e-18 | 69 | 26 | 5 |

## 6. Conclusion

As a combined routing and scheduling problem, the Traveling Salesman Problem with Job-time has a number of relevant practical applications in real life. This paper introduced a breakout local search algorithm, which employs a tabu search to explore two dedicated neighborhoods and applies a combined perturbation to escape local optima. A neighborhood reduction strategy was designed to identify promising neighbor solutions and accelerate the search process.

The proposed algorithm has been assessed on four sets of 310 instances in the literature and showed a highly competitive performance compared to the current best methods. Specifically, our algorithm has established new best-known results (updated upper bounds) for 291 out of the 310 benchmark instances ($> 93\%$ cases). Additional experiments have confirmed the usefulness of the neighborhood reduction and combined perturbation for the performance of the algorithm.

The algorithm in this work can be further improved. First, since the TSPJ involves two classic NP-hard problems, the search space is extremely complex. It is worth investigating other neighborhoods based on dedicated features of the problem to be able to explore the space more effectively. Second, the algorithm needs to make decisions during its search process (e.g., when should the perturbation be triggered, which type of perturbation should be applied...). To ensure informative decisions, reinforcement learning techniques could be useful. Finally, no dedicated exact algorithm exists for the problem studied in this work. Research on exact algorithms is thus needed.

### Acknowledgments

We thank the authors of (Mosayebi et al., 2021) for making the TSPJ benchmark instances available and answering our questions. This work was partially supported by the National Natural Science Foundation Program of China (Grant No. 72122006). Support from CSC (Grant No. 202106050037) for the first author is acknowledged.

### References

Aksan, Y., Dökeroglu, T., & Cosar, A. (2017). A stagnation-aware cooperative parallel breakout local search algorithm for the quadratic assignment problem. *Computers & Industrial Engineering*, *103*, 105–115.

Barbarosoğlu, G., Özdamar, L., & Cevik, A. (2002). An interactive approach for hierarchical analysis of helicopter logistics in disaster relief operations. *European Journal of Operational Research*, *140*, 118–133.

Basnet, C. B., Foulds, L. R., & Wilson, J. M. (2006). Scheduling contractors' farm-to-farm crop harvesting operations. *International Transactions in Operational Research*, *13*, 1–15.

Bays, M. J., & Wettergren, T. A. (2017). Service agent–transport agent task planning incorporating robust scheduling techniques. *Robotics and Autonomous Systems*, *89*, 15–26.

Benlic, U., & Hao, J.-K. (2013a). Breakout local search for maximum clique problems. *Computers & Operations Research*, *40*, 192–206.

Benlic, U., & Hao, J.-K. (2013b). Breakout local search for the max-cut problem. *Engineering Applications of Artificial Intelligence*, *26*, 1162–1173.

Benlic, U., & Hao, J.-K. (2013c). Breakout local search for the quadratic assignment problem. *Applied Mathematics and Computation*, *219*, 4800–4815.

Benlic, U., & Hao, J.-K. (2015). Memetic search for the quadratic assignment problem. *Expert Systems with Applications*, *42*, 584–595.

Das, S. K., & Nagendra, P. (1997). Selection of routes in a flexible manufacturing facility. *International Journal of Production Economics*, *48*, 237–247.

Fu, Z.-H., & Hao, J.-K. (2014). Breakout local search for the steiner tree problem with revenue, budget and hop constraints. *European Journal of Operational Research*, *232*, 209–220.

Gavish, B., & Graves, S. C. (1978). The travelling salesman problem and related problems. *Working paper*, *GR-078-78*, Operations Research Center, Massachusetts Institute of Techno.

Ghandi, S., & Masehian, E. (2015). A breakout local search (BLS) method for solving the assembly sequence planning problem. *Engineering Applications of Artificial Intelligence*, *39*, 245–266.

Glover, F., & Laguna, M. (1998). Tabu search. In *Handbook of Combinatorial Optimization* (pp. 2093–2229). Springer.

Hoos, H. H., & Stützle, T. (2004). *Stochastic Local Search: Foundations & Applications*. Elsevier / Morgan Kaufmann.

Krari, M. E., Ahiod, B., & Benani, B. E. (2018). Breakout local search for the travelling salesman problem. *Computing and Informatics*, *37*, 656–672.

Lahrichi, Y., Deroussi, L., Grangeon, N., & Norre, S. (2020). A min-max path approach for balancing robotic assembly lines with sequence-dependent setup times. In *Proceedings of 13ème Conference Internationale de Modlisation, Optimisation et Simulation (MOSIM 2020), 12-14 November, Agadir, Marocco*.

Li, M., Hao, J.-K., & Wu, Q. (2020). General swap-based multiple neighborhood adaptive search for the maximum balanced biclique problem. *Computers & Operations Research*, *119*, 104922.

Lin, S. (1965). Computer solutions of the traveling salesman problem. *Bell System Technical Journal*, *44*, 2245–2269.

López-Ibáñez, M., Dubois-Lacoste, J., Cáceres, L. P., Birattari, M., & Stützle, T. (2016). The irace package: Iterated racing for automatic algorithm configuration. *Operations Research Perspectives*, *3*, 43–58.

Lourenço, H. R., Martin, O. C., & Stützle, T. (2003). Iterated local search. In *Handbook of Metaheuristics* (pp. 320–353). Springer.

Mosayebi, M., Sodhi, M., & Wettergren, T. A. (2021). The traveling salesman problem with job-times (tspj). *Computers & Operations Research*, *129*, 105226.

Rashidnejad, M., Ebrahimnejad, S., & Safari, J. (2018). A bi-objective model of preventive maintenance planning in distributed systems considering vehicle routing problem. *Computers & Industrial Engineering*, *120*, 360–381.

Reinelt, G. (1991). Tsplib—a traveling salesman problem library. *ORSA Journal on Computing*, *3*, 376–384.

Taillard, É. D. (1991). Robust taboo search for the quadratic assignment problem. *Parallel Computing*, *17*, 443–455.

Zhang, L., Deng, Q., Lin, R., Gong, G., & Han, W. (2021). A combinatorial evolutionary algorithm for unrelated parallel machine scheduling problem with sequence and machine-dependent setup times, limited worker resources and learning effect. *Expert Systems with Applications*, *175*, 114843.

Zhou, Y., & Hao, J.-K. (2017). Frequency-driven tabu search for the maximum s-plex problem. *Computers & Operations Research*, *86*, 65–78.

## Appendix A. Mathematical model of the TSPJ (Mosayebi et al., 2021)

The mathematical model of the TSPJ presented in (Mosayebi et al., 2021) is based on the classical TSP model proposed by Gavish & Graves (1978). The TSPJ can be defined on an complete graph $G = (L, E)$ and a job set $J$. Let $E$ be a set of edges, and $L = (l_0, l_1, \ldots, l_n)$ be the set of location nodes with the vertex 0 being the depot. Some other notions used in the model are:

**Parameters:**

$Y_{lk}$: The sequence number of edges visited;

$C_{max}$: The maximum completion time among all the jobs;

$TS_l$: The arriving time at node $l$, also the time to start work of the job assigned to this node;

$D_{lk}$: The travel-time from node $l$ to node $k$;

$jt_{lj}$: The processing time of job $j$ assigned to node $l$;

**Variables:**

$X_{lk}$: Indicating whether the edges $E_{lk}$ is traversed from node $l$ to node $k$;

$Z_{lj}$: Indicating whether the job $j$ is assigned to the location node $l$;

On the basis of the above mentioned parameters and decision variables, the TSPJ is formulated as the following mixed integer program:

$$\min \quad C_{max} \tag{A.1}$$

$$\text{s.t.} \quad C_{max} \geq TS_l + \sum_{j=1}^{n} Z_{lj} jt_{lj} \quad \forall l = 1, \ldots, n \tag{A.2}$$

$$C_{max} \geq TS_l + Z_{l0} jt_{l0} \quad \forall l = 1, \ldots, n \tag{A.3}$$

$$\sum_{l=1}^{n} Z_{lj} = 1 \quad \forall j = 1, \ldots, n \tag{A.4}$$

$$\sum_{j=1}^{n} Z_{lj} = 1 \quad \forall l = 1, \ldots, n \tag{A.5}$$

$$\sum_{k=0}^{n} X_{lk} = 1 \quad \forall l = 0, \ldots, n \quad l \neq k \tag{A.6}$$

$$\sum_{l=0}^{n} X_{lk} = 1 \quad \forall k = 0, \ldots, n \quad l \neq k \tag{A.7}$$

$$\sum_{k=0}^{n} Y_{lk} - \sum_{k=0}^{n} Y_{kl} = 1 \quad \forall l = 1, \ldots, n \quad l \neq k \tag{A.8}$$

$$Y_{lk} \leq n X_{lk} \quad \forall l = 1, \ldots, n \quad \forall k = 0, \ldots, n \quad l \neq k \tag{A.9}$$

$$TS_l + D_{lk} - (1 - X_{lk}) M \leq TS_k \quad \forall l = 0, \ldots, n \quad \forall k = 1, \ldots, n \quad l \neq k \tag{A.10}$$

$$X_{lk}, Z_{lj} \in \{0, 1\}, \quad TS_l, Y_{lk} \geq 0 \quad \forall l = 1, \ldots, n \quad k = 0, \ldots, n \quad j = 1, \ldots, n \tag{A.11}$$

Eqs. (A.1)-(A.3) are used to calculate the objective value. Eqs. (A.4) and (A.5) are the job assignment constraints, which require that one job is assigned to one location node and vice versa. Eqs. (A.6) and (Eqs.A.7) are the route restrictions that force that all the nodes are visited only once. Eqs. (A.8) and (Eqs.A.9) are the subtour eliminators. Eq. (A.10) is the job processing start time restriction between different location nodes; the node arriving time (i.e., the job processing start time) cannot be less than the arriving time of the previous location node plus the traveling time between them. $M$ is a large enough number. Eq. 6(A.11) defines the domain of each variable $X$,$Z$,$Y$ and $TS$.

## Appendix B. Detailed results

This section shows detailed computational results of the proposed BLS algorithm compared to the results of the reference algorithms in (Mosayebi et al., 2021) on the four sets of TSPJ benchmark instances. The results of the CPLEX

MIP solver with the model of Appendix A with a time limit of 7200 seconds are also included for the (small) instances of Set I.

Table B.6 shows the results on the 10 instances of Set I. In this table, column 1 gives the name of instances, columns 2 and 3 show the upper bound and lower bound obtained by GAMS/CPLEX, columns 4 and 5 show the best results among the reference algorithms and the time needed to reach each result. The data in columns 2 - 5 are directly extracted from (Mosayebi et al., 2021). Columns 6 and 7 give the results obtained by our BLS algorithm under the cutoff conditions given in Section 4.2. gap-1 and gap-2 in columns 8 and 9 are the gaps between our results with respect to the best upper bound obtained by CPLEX and the reference algorithms, respectively. A negative gap (in bold) indicates an improved upper bound. We observe that BLS can find improved upper bounds for all instances except one case.

Table B.7 shows the results on the 100 instances of Set II. Column 1 is the name of the instance, columns 2 - 3 show the best results among the four reference algorithms of (Mosayebi et al., 2021) and the times needed to attain these best results. Columns 4 - 7 are the results of our BLS algorithm according to two stopping conditions (see Section 4.2) and the time to get these results. BLS-1 shows the results under the cutoff condition of (Mosayebi et al., 2021) (0.0012 seconds for Set II). BLS-2 shows the results under the relaxed cutoff condition (30 seconds for Set II). Columns 8 and 9 show the upper and lower bound from the CPLEX MIP solver using the model of Appendix A. Columns 10 - 11 show the gaps between BLS with the best-known results. A negative gap (in bold) indicates an improved upper bound. We observe that BLS can find improved upper bounds for all instances except 17 cases.

Tables B.8 and B.9 shows the results on the 200 instances of sets III and IV with the same information as in Table B.7 except that the results of CPLEX are not reported due to the fact that CPLEX reports within 7200 seconds bad results for the instances of Set III and even fails to find a feasible solution for the instances of Set IV. From these results, we observe that BLS is able to improve all previous upper bounds for the 200 instances of sets III and IV except 2 cases.

Finally, Tables B.10 - B.12 show the best results and the times to get these results of the four re-implemented algorithms of (Mosayebi et al., 2021) on each instance of Set II, Set III, and Set IV, respectively. These results were obtained under the cutoff condition of (Mosayebi et al., 2021). We observe that compared to the results reported in (Mosayebi et al., 2021), our reimplementation obtains slightly better results for Set II and better results for sets III and IV.

Table B.6: Computational results of the proposed BLS algorithm and comparison with the best-known results from the four references of (Mosayebi et al., 2021) on instances from Set I.

| Instance | CPLEX | | Reference algorithm | | BLS | | gap-1 | gap-2 |
|---|---|---|---|---|---|---|---|---|
| | $UB$ | $LB$ | $f_{bks}$ | $t_{bks}$ | $f_{best}$ | $t_{best}$ | | |
| gr12-J | 2760 | 2760 | 2760 | 0.13 | 2760 | 0.0001 | 0 | 0 |
| gr21-J | 7788 | 7712 | 7956 | 0.21 | 7788 | 0.001 | 0 | -2.11 |
| gr24-J | 1806 | 1802 | 1818 | 0.34 | 1806 | 0.001 | 0 | -0.66 |
| fri26-J | 1283 | 1282.94 | 1326 | 0.22 | 1283 | 0.001 | 0 | -3.24 |
| bays29-J | 2937 | 2892.88 | 2940 | 0.57 | **2916** | 0.001 | -0.715 | -0.82 |
| gr48-J | 7288 | 7215.36 | 7499 | 2.48 | **7282** | 0.019 | -0.0823 | -2.89 |
| eil51-J | 630 | 627.94 | 640.2 | 5.69 | **628.51** | 0.014 | -0.2365 | -1.83 |
| berlin52-J | 11087.5 | 10976.96 | 11225.77 | 1.41 | **11087.21** | 0.001 | -0.0026 | -1.23 |
| eil76-J | 802.27 | 799.47 | 822.46 | 3.32 | **801.91** | 6.26 | -0.0449 | -2.5 |
| eil101-J | 947.42 | 940.59 | 975.94 | 14.75 | **946.33** | 59.34 | -0.1151 | -3.03 |

Table B.7: Computational results of the proposed BLS algorithm and comparison with the best results from the four references of (Mosayebi et al., 2021) on the instances from Set II.

| Instance | Reference algorithms | | BLS-1 | | BLS-2 | | UB | LB | gap-1 | gap-2 |
|---|---|---|---|---|---|---|---|---|---|---|
| | $f_{bks}$ | $t_{bks}$ | $f_{best}$ | $t_{best}$ | $f_{best}$ | $t_{best}$ | | | | |
| 1 | 319 | 0.00114 | 301 | 0.00114 | 301 | 0.00218 | 367 | 125 | -5.64 | -5.64 |
| 2 | 273 | 0.00116 | 279 | 0.00096 | 267 | 0.00156 | 300 | 106 | 2.2 | -2.2 |
| 3 | 288 | 0.00117 | 285 | 0.001 | **280** | 0.00263 | 319 | 120 | -1.04 | -2.78 |
| 4 | 289 | 0.00189 | 291 | 0.00107 | **283** | 0.00175 | 376 | 114 | 0.69 | -2.08 |
| 5 | 282 | 0.00247 | 281 | 0.00114 | **280** | 0.00343 | 356 | 118 | -0.35 | -0.71 |
| 6 | 305 | 0.00063 | 303 | 0.00084 | **299** | 0.00086 | 347 | 127 | -0.66 | -1.97 |
| 7 | 286 | 0.00254 | 288 | 0.00116 | **285** | 0.00212 | 332 | 113 | 0.7 | -0.35 |
| 8 | 254 | 0.00071 | 257 | 0.001 | **252** | 0.00233 | 300 | 104 | 1.18 | -0.79 |
| 9 | 290 | 0.00066 | 287 | 0.0007 | **285** | 0.0019 | 313 | 121.21 | -1.03 | -1.72 |
| 10 | 313 | 0.00167 | 314 | 0.00113 | 313 | 0.00185 | 379 | 127 | 0.32 | 0 |
| 11 | 299 | 0.00079 | 303 | 0.00101 | **297** | 0.00245 | 334 | 121 | 1.34 | -0.67 |
| 12 | 284 | 0.00322 | 282 | 0.00098 | **280** | 0.00274 | 338 | 124 | -0.7 | -1.41 |
| 13 | 268 | 0.00127 | 268 | 0.00124 | **263** | 0.00177 | 336 | 105 | 0 | -1.87 |
| 14 | 270 | 0.00107 | 267 | 0.00056 | 267 | 0.00078 | 335 | 105 | -1.11 | -1.11 |
| 15 | 271 | 0.00056 | 271 | 0.00115 | **270** | 0.00153 | 313 | 111.82 | 0 | -0.37 |
| 16 | 299 | 0.00125 | 303 | 0.00113 | **297** | 0.00327 | 371 | 127 | 1.34 | -0.67 |
| 17 | 265 | 0.001 | 270 | 0.00106 | **263** | 0.00267 | 317 | 112 | 1.89 | -0.75 |
| 18 | 258 | 0.00082 | 252 | 0.00093 | 252 | 0.00096 | 286 | 100.15 | -2.33 | -2.33 |
| 19 | 248 | 0.00076 | 241 | 0.00105 | **239** | 0.00195 | 264 | 98.67 | -2.82 | -3.63 |
| 20 | 289 | 0.0006 | 283 | 0.00114 | 283 | 0.00119 | 369 | 120 | -2.08 | -2.08 |
| 21 | 285 | 0.00089 | 291 | 0.00104 | **284** | 0.00093 | 329 | 117 | 2.11 | -0.35 |
| 22 | 236 | 0.00078 | 242 | 0.00109 | 236 | 0.00302 | 257 | 99 | 2.54 | 0 |
| 23 | 279 | 0.0007 | 283 | 0.00103 | **276** | 0.00163 | 313 | 108 | 1.43 | -1.08 |
| 24 | 298 | 0.00099 | 295 | 0.00115 | **292** | 0.00206 | 376 | 121 | -1.01 | -2.01 |
| 25 | 240 | 0.00074 | 248 | 0.00085 | **238** | 0.00132 | 273 | 97 | 3.33 | -0.83 |
| 26 | 291 | 0.00076 | 298 | 0.00117 | 291 | 0.00338 | 356 | 125 | 2.41 | 0 |
| 27 | 297 | 0.00184 | 297 | 0.0011 | **296** | 0.00298 | 403 | 131 | 0 | -0.34 |
| 28 | 285 | 0.00191 | 283 | 0.00119 | 283 | 0.0012 | 331 | 117 | -0.7 | -0.7 |
| 29 | 299 | 0.00232 | 299 | 0.00092 | 299 | 0.00154 | 328 | 132 | 0 | 0 |
| 30 | 304 | 0.00205 | 300 | 0.00121 | **298** | 0.00252 | 369 | 125 | -1.32 | -1.97 |
| 31 | 261 | 0.00083 | 259 | 0.00121 | 259 | 0.00172 | 314 | 114 | -0.77 | -0.77 |
| 32 | 325 | 0.00099 | 319 | 0.00108 | 319 | 0.00116 | 362 | 137.26 | -1.85 | -1.85 |
| 33 | 306 | 0.00181 | 306 | 0.00112 | 306 | 0.00186 | 359 | 122 | 0 | 0 |
| 34 | 294 | 0.00154 | 294 | 0.00119 | 294 | 0.00144 | 337 | 128 | 0 | 0 |
| 35 | 283 | 0.00184 | 280 | 0.00118 | 280 | 0.00136 | 314 | 120 | -1.06 | -1.06 |
| 36 | **263** | 0.0016 | 264 | 0.00078 | 264 | 0.00078 | 299 | 119 | 0.38 | 0.38 |
| 37 | 279 | 0.00082 | 271 | 0.00099 | 271 | 0.0015 | 294 | 111 | -2.87 | -2.87 |
| 38 | 249 | 0.00064 | 250 | 0.00112 | **247** | 0.00104 | 295 | 105 | 0.4 | -0.8 |
| 39 | 276 | 0.00086 | 279 | 0.00089 | **274** | 0.0007 | 348 | 126 | 1.09 | -0.72 |
| 40 | 256 | 0.00116 | 258 | 0.0009 | **255** | 0.00206 | 293 | 107 | 0.78 | -0.39 |
| 41 | 287 | 0.00052 | 291 | 0.00088 | **282** | 0.00127 | 303 | 121 | 1.39 | -1.74 |
| 42 | 297 | 0.00122 | 295 | 0.00101 | **292** | 0.00235 | 328 | 116.26 | -0.67 | -1.68 |
| 43 | 297 | 0.00129 | 294 | 0.00109 | 294 | 0.00175 | 325 | 124 | -1.01 | -1.01 |
| 44 | 271 | 0.0011 | 267 | 0.00111 | 267 | 0.00139 | 301 | 109 | -1.48 | -1.48 |
| 45 | 325 | 0.00163 | 327 | 0.00093 | **322** | 0.00386 | 437 | 133 | 0.62 | -0.92 |
| 46 | 248 | 0.00066 | 246 | 0.00107 | **242** | 0.0011 | 256 | 103 | -0.81 | -2.42 |
| 47 | 325 | 0.00086 | 319 | 0.00119 | **317** | 0.00128 | 368 | 133 | -1.85 | -2.46 |
| 48 | 300 | 0.00145 | 299 | 0.0011 | **295** | 0.0018 | 330 | 124 | -0.33 | -1.67 |
| 49 | 265 | 0.00105 | 266 | 0.00108 | **264** | 0.00156 | 337 | 115.99 | 0.38 | -0.38 |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 50 | 293 | 0.00195 | 292 | 0.00094 | **291** | 0.00349 | 325 | 125 | -0.34 | -0.68 |
| 51 | 275 | 0.00111 | 275 | 0.00091 | 275 | 0.00106 | 316 | 118 | 0 | 0 |
| 52 | 341 | 0.00155 | 338 | 0.00103 | 338 | 0.00215 | 412 | 138 | -0.88 | -0.88 |
| 53 | 286 | 0.00124 | 286 | 0.00093 | **285** | 0.00248 | 349 | 117 | 0 | -0.35 |
| 54 | 267 | 0.00071 | 269 | 0.00096 | **265** | 0.00172 | 307 | 114 | 0.75 | -0.75 |
| 55 | 304 | 0.0005 | 305 | 0.00097 | **300** | 0.00226 | 342 | 130 | 0.33 | -1.32 |
| 56 | 289 | 0.00084 | 292 | 0.00067 | **288** | 0.00079 | 339 | 127 | 1.04 | -0.35 |
| 57 | 257 | 0.00154 | 257 | 0.00113 | **254** | 0.00257 | 305 | 111 | 0 | -1.17 |
| 58 | 341 | 0.00134 | 336 | 0.00079 | 336 | 0.00113 | 370 | 142 | -1.47 | -1.47 |
| 59 | 288 | 0.00107 | 285 | 0.0009 | **284** | 0.00205 | 371 | 118 | -1.04 | -1.39 |
| 60 | 257 | 0.001 | 256 | 0.00086 | 256 | 0.00286 | 308 | 100 | -0.39 | -0.39 |
| 61 | 307 | 0.00187 | 313 | 0.0011 | 307 | 0.00258 | 366 | 131 | 1.95 | 0 |
| 62 | 289 | 0.00162 | 289 | 0.00101 | 289 | 0.00117 | 332 | 120 | 0 | 0 |
| 63 | 274 | 0.00145 | 274 | 0.00094 | 274 | 0.00111 | 377 | 113 | 0 | 0 |
| 64 | 290 | 0.00082 | 309 | 0.00098 | **283** | 0.00385 | 289 | 111.06 | 6.55 | -2.41 |
| 65 | 269 | 0.00112 | 268 | 0.00076 | 268 | 0.00115 | 422 | 118 | -0.37 | -0.37 |
| 66 | 290 | 0.00188 | 287 | 0.00121 | **286** | 0.00302 | 482 | 132 | -1.03 | -1.38 |
| 67 | 320 | 0.00121 | 325 | 0.00117 | **319** | 0.00225 | 331 | 126 | 1.56 | -0.31 |
| 68 | 301 | 0.00179 | 299 | 0.00133 | **295** | 0.00366 | 369 | 121 | -0.66 | -1.99 |
| 69 | 300 | 0.00226 | 311 | 0.00115 | **297** | 0.00232 | 299 | 114 | 3.67 | -1 |
| 70 | 273 | 0.00123 | 273 | 0.00078 | 273 | 0.00081 | 299 | 114 | 0 | 0 |
| 71 | 295 | 0.00123 | 295 | 0.00086 | 295 | 0.00085 | 384 | 118 | 0 | 0 |
| 72 | 259 | 0.00091 | 256 | 0.00093 | 256 | 0.00078 | 271 | 107 | -1.16 | -1.16 |
| 73 | 278 | 0.00177 | 277 | 0.00087 | **275** | 0.00154 | 320 | 119 | -0.36 | -1.08 |
| 74 | 271 | 0.00088 | 271 | 0.00116 | 271 | 0.00106 | 305 | 113 | 0 | 0 |
| 75 | 212 | 0.00063 | 212 | 0.00075 | 212 | 0.00051 | 259 | 88 | 0 | 0 |
| 76 | 297 | 0.00084 | 294 | 0.00088 | **293** | 0.00101 | 321 | 120 | -1.01 | -1.35 |
| 77 | 274 | 0.00109 | 273 | 0.00095 | **271** | 0.00177 | 313 | 122 | -0.36 | -1.09 |
| 78 | 258 | 0.00099 | 261 | 0.00109 | **255** | 0.00268 | 291 | 111 | 1.16 | -1.16 |
| 79 | 281 | 0.0013 | 280 | 0.00104 | 280 | 0.00101 | 315 | 113 | -0.36 | -0.36 |
| 80 | 280 | 0.00079 | 281 | 0.00091 | **271** | 0.00244 | 299 | 117 | 0.36 | -3.21 |
| 81 | 259 | 0.00148 | 261 | 0.00093 | 259 | 0.00115 | 284 | 112 | 0.77 | 0 |
| 82 | 345 | 0.00187 | 348 | 0.00115 | **339** | 0.0016 | 482 | 146 | 0.87 | -1.74 |
| 83 | 259 | 0.00175 | 257 | 0.00109 | 257 | 0.00118 | 341 | 104 | -0.77 | -0.77 |
| 84 | 272 | 0.00108 | 267 | 0.00056 | **264** | 0.00195 | 274 | 114 | -1.84 | -2.94 |
| 85 | 300 | 0.00124 | 300 | 0.00076 | **296** | 0.00267 | 333 | 123 | 0 | -1.33 |
| 86 | 338 | 0.00063 | 342 | 0.00073 | **333** | 0.00232 | 374 | 139 | 1.18 | -1.48 |
| 87 | 261 | 0.00087 | 261 | 0.0012 | **257** | 0.00223 | 303 | 107 | 0 | -1.53 |
| 88 | 305 | 0.00169 | 299 | 0.00098 | **295** | 0.00281 | 328 | 126 | -1.97 | -3.28 |
| 89 | 288 | 0.00084 | 289 | 0.00069 | **287** | 0.00098 | 302 | 113 | 0.35 | -0.35 |
| 90 | 322 | 0.00108 | 327 | 0.00119 | **321** | 0.00141 | 346 | 127 | 1.55 | -0.31 |
| 91 | 281 | 0.00115 | 280 | 0.00098 | **274** | 0.00302 | 360 | 115 | -0.36 | -2.49 |
| 92 | 300 | 0.00061 | 309 | 0.00071 | **296** | 0.00249 | 389 | 130 | 3 | -1.33 |
| 93 | 309 | 0.0012 | 299 | 0.00104 | **295** | 0.00218 | 312 | 120 | -3.24 | -4.53 |
| 94 | 309 | 0.00105 | 308 | 0.0012 | **307** | 0.00302 | 323 | 123 | -0.32 | -0.65 |
| 95 | 260 | 0.00126 | 256 | 0.0012 | 256 | 0.00129 | 269 | 103 | -1.54 | -1.54 |
| 96 | 268 | 0.00092 | 273 | 0.0012 | **263** | 0.00428 | 273 | 106 | 1.87 | -1.87 |
| 97 | 256 | 0.00083 | 249 | 0.00074 | 249 | 0.00125 | 259 | 99 | -2.73 | -2.73 |
| 98 | 298 | 0.00068 | 298 | 0.00119 | **289** | 0.00192 | 304 | 116 | 0 | -3.02 |
| 99 | 278 | 0.00135 | 277 | 0.0012 | **265** | 0.00171 | 277 | 106 | -0.36 | -4.68 |
| 100 | 249 | 0.00083 | 248 | 0.00119 | 248 | 0.00086 | 263 | 99 | -0.4 | -0.4 |
| Avg | 284.44 | 0.00121 | 284.33 | 0.00101 | **280.85** | 0.00189 | - | - | -0.0297 | -1.2484 |

Table B.8: Computational results of the proposed BLS algorithm and comparison with the best results from the four references of (Mosayebi et al., 2021) on the instances of Set III.

| Instance | Reference algorithms | | BLS-1 | | BLS-2 | | gap-1 | gap-2 |
|---|---|---|---|---|---|---|---|---|
| | $f_{bks}$ | $t_{bks}$ | $f_{best}$ | $t_{best}$ | $f_{best}$ | $t_{best}$ | | |
| 1 | 3235 | 0.71 | 3219 | 2.18 | **3187** | 2.64 | -0.49 | -1.48 |
| 2 | 3494 | 0.57 | 3452 | 0.91 | **3442** | 2.09 | -1.2 | -1.49 |
| 3 | 3467 | 3.59 | 3436 | 1.59 | **3426** | 3.36 | -0.89 | -1.18 |
| 4 | 3465 | 4.44 | 3447 | 0.81 | 3447 | 1.1 | -0.52 | -0.52 |
| 5 | 3673 | 0.66 | 3647 | 1.94 | **3637** | 3.47 | -0.71 | -0.98 |
| 6 | 3273 | 0.61 | **3233** | 1.12 | 3237 | 0.97 | -1.22 | -1.1 |
| 7 | 3571 | 0.93 | 3537 | 1.72 | **3521** | 5.09 | -0.95 | -1.4 |
| 8 | 3604 | 0.68 | 3562 | 2.06 | **3558** | 3.72 | -1.17 | -1.28 |
| 9 | 3335 | 0.76 | 3330 | 2.07 | **3320** | 2.31 | -0.15 | -0.45 |
| 10 | 3544 | 0.89 | 3514 | 1.6 | **3497** | 2.48 | -0.85 | -1.33 |
| 11 | 3401 | 0.73 | **3362** | 2.18 | 3363 | 1.38 | -1.15 | -1.12 |
| 12 | 3510 | 1.88 | 3473 | 1.63 | **3457** | 2.87 | -1.05 | -1.51 |
| 13 | 3523 | 1.14 | 3514 | 1.96 | **3476** | 4.63 | -0.26 | -1.33 |
| 14 | 3480 | 0.42 | 3456 | 1.09 | **3444** | 1.38 | -0.69 | -1.03 |
| 15 | 3462 | 2.04 | 3444 | 1.26 | 3444 | 2.03 | -0.52 | -0.52 |
| 16 | 3699 | 7.22 | 3661 | 1.87 | **3655** | 3.59 | -1.03 | -1.19 |
| 17 | 3571 | 7.89 | 3502 | 1.79 | **3491** | 4.05 | -1.93 | -2.24 |
| 18 | 3593 | 0.99 | 3572 | 1.2 | **3558** | 2.7 | -0.58 | -0.97 |
| 19 | 3501 | 3.68 | **3432** | 1.77 | 3438 | 1.3 | -1.97 | -1.8 |
| 20 | 3504 | 0.72 | 3476 | 0.88 | 3476 | 1.3 | -0.8 | -0.8 |
| 21 | 3589 | 0.5 | 3564 | 2.08 | **3561** | 2.24 | -0.7 | -0.78 |
| 22 | 3685 | 9.75 | 3654 | 1.58 | **3649** | 3.52 | -0.84 | -0.98 |
| 23 | 3398 | 2.66 | 3354 | 1.34 | **3349** | 2.29 | -1.29 | -1.44 |
| 24 | 3476 | 2.41 | 3455 | 2.07 | **3412** | 10.41 | -0.6 | -1.84 |
| 25 | 3671 | 7.15 | 3617 | 2 | 3606 | **6.36** | -1.47 | -1.77 |
| 26 | 3477 | 0.73 | 3433 | 1.9 | 3433 | 2.49 | -1.27 | -1.27 |
| 27 | 3576 | 0.9 | 3584 | 2.19 | **3554** | 2.66 | 0.22 | -0.62 |
| 28 | 3505 | 0.82 | **3456** | 2.03 | 3467 | 1.83 | -1.4 | -1.08 |
| 29 | 3327 | 0.53 | 3269 | 0.85 | 3269 | 0.9 | -1.74 | -1.74 |
| 30 | 3255 | 1.69 | 3215 | 0.93 | **3210** | 3.37 | -1.23 | -1.38 |
| 31 | 3324 | 0.37 | 3284 | 2.08 | **3256** | 1.39 | -1.2 | -2.05 |
| 32 | 3559 | 0.91 | 3563 | 1.62 | **3538** | 2.57 | 0.11 | -0.59 |
| 33 | 3667 | 0.96 | 3615 | 2.05 | **3612** | 1.63 | -1.42 | -1.5 |
| 34 | 3492 | 5.32 | 3434 | 1.29 | **3424** | 2.9 | -1.66 | -1.95 |
| 35 | 3505 | 1.87 | 3429 | 1.49 | **3427** | 3.06 | -2.17 | -2.23 |
| 36 | 3467 | 0.4 | **3433** | 1.54 | 3438 | 1.46 | -0.98 | -0.84 |
| 37 | 3663 | 4.94 | 3651 | 1.83 | 3651 | 4.36 | -0.33 | -0.33 |
| 38 | 3621 | 0.91 | 3581 | 1.72 | **3578** | 2.82 | -1.1 | -1.19 |
| 39 | 3432 | 0.53 | 3390 | 1.75 | **3387** | 2.66 | -1.22 | -1.31 |
| 40 | 3293 | 0.35 | 3250 | 1.11 | **3248** | 1.31 | -1.31 | -1.37 |
| 41 | 3465 | 0.8 | 3429 | 1.05 | **3419** | 2.66 | -1.04 | -1.33 |
| 42 | 3506 | 0.79 | 3479 | 2.18 | **3471** | 2.87 | -0.77 | -1 |
| 43 | 3406 | 1.68 | 3390 | 1.56 | **3383** | 1.95 | -0.47 | -0.68 |
| 44 | 3616 | 6.17 | 3550 | 2.02 | **3548** | 5 | -1.83 | -1.88 |
| 45 | 3587 | 2.61 | **3529** | 1.48 | 3526 | 2.52 | -1.62 | -1.7 |
| 46 | 3386 | 2.27 | 3347 | 1.46 | **3333** | 0.96 | -1.15 | -1.57 |
| 47 | 3335 | 0.69 | 3306 | 0.97 | **3290** | 3.14 | -0.87 | -1.35 |
| 48 | **3558** | 0.78 | 3559 | 2.15 | 3559 | 1.98 | -0.25 | 0.03 |
| 49 | 3602 | 0.83 | **3550** | 2.02 | 3562 | 1.73 | -1.44 | -1.11 |
| 50 | 3594 | 8.24 | 3529 | 1.85 | **3526** | 3.15 | -1.81 | -1.89 |
| 51 | 3373 | 0.52 | 3336 | 0.72 | 3336 | 1.14 | -1.1 | -1.1 |
| 52 | 3317 | 0.67 | 3299 | 1.45 | **3293** | 4.29 | -0.54 | -0.72 |
| 53 | 3528 | 0.44 | **3473** | 2.08 | 3480 | 1.14 | -1.56 | -1.36 |
| 54 | 3368 | 7.44 | 3347 | 1.07 | **3340** | 2.23 | -0.62 | -0.83 |
| 55 | 3739 | 0.57 | 3677 | 2.06 | 3677 | 1.56 | -1.66 | -1.66 |
| 56 | 3682 | 0.51 | 3648 | 2.16 | **3626** | 4.59 | -0.92 | -1.52 |
| 57 | 3709 | 8.86 | 3671 | 1.88 | **3665** | 3.73 | -1.02 | -1.19 |
| 58 | 3633 | 0.63 | 3591 | 1.85 | 3591 | 2.47 | -1.16 | -1.16 |
| 59 | 3302 | 0.4 | 3250 | 0.93 | **3244** | 1.12 | -1.57 | -1.76 |
| 60 | 3392 | 3.37 | 3364 | 0.85 | 3364 | 0.94 | -0.83 | -0.83 |
| 61 | 3421 | 3.92 | 3367 | 1.48 | **3365** | 1.6 | -1.58 | -1.64 |
| 62 | 3609 | 0.74 | 3574 | 1.61 | **3567** | 2.03 | -0.97 | -1.16 |
| 63 | 3558 | 0.51 | 3532 | 1.67 | **3526** | 3.35 | -0.73 | -0.9 |
| 64 | 3470 | 2.03 | 3452 | 1.16 | 3452 | 1.27 | -0.52 | -0.52 |
| 65 | 3732 | 1.01 | 3691 | 1.5 | **3687** | 2.11 | -1.1 | -1.21 |
| 66 | 3645 | 4.25 | 3610 | 1.81 | **3603** | 3.35 | -0.96 | -1.15 |
| 67 | 3427 | 2.38 | 3421 | 2.05 | **3420** | 2.14 | -0.18 | -0.2 |
| 68 | 3343 | 2.11 | 3303 | 1.61 | **3296** | 2.07 | -1.2 | -1.41 |
| 69 | 3641 | 0.56 | 3598 | 1.9 | **3568** | 3.61 | -1.18 | -2 |
| 70 | 3520 | 0.89 | 3495 | 1.16 | **3482** | 2.08 | -0.71 | -1.08 |
| 71 | 3599 | 0.76 | 3564 | 1 | **3563** | 2.04 | -0.97 | -1 |
| 72 | 3623 | 5.24 | 3576 | 2.12 | **3558** | 2.63 | -1.3 | -1.79 |
| 73 | 3413 | 4.51 | 3339 | 1.88 | **3335** | 2.25 | -2.17 | -2.29 |
| 74 | 3281 | 0.6 | **3252** | 1.86 | 3255 | 1.61 | -0.88 | -0.79 |
| 75 | 3390 | 3.65 | 3368 | 0.83 | **3358** | 2.25 | -0.65 | -0.94 |
| 76 | 3645 | 0.85 | 3601 | 1.7 | **3595** | 2.87 | -1.21 | -1.37 |
| 77 | 3472 | 0.56 | 3445 | 1.62 | 3445 | 1.91 | -0.78 | -0.78 |
| 78 | 3444 | 3.48 | 3421 | 0.95 | 3421 | 1.07 | -0.67 | -0.67 |
| 79 | 3676 | 1.07 | 3646 | 2.14 | **3608** | 3.25 | -0.82 | -1.85 |
| 80 | 3554 | 0.76 | 3535 | 1.87 | **3528** | 7.22 | -0.53 | -0.73 |
| 81 | 3437 | 3.66 | 3381 | 0.64 | 3381 | 0.8 | -1.63 | -1.63 |
| 82 | 3475 | 1.92 | 3429 | 1.89 | **3419** | 4.39 | -1.32 | -1.61 |
| 83 | 3428 | 0.73 | **3364** | 2.08 | 3369 | 1.86 | -1.87 | -1.72 |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 84 | 3567 | 4.36 | 3523 | 1.98 | **3503** | 2.07 | -1.23 | -1.79 |
| 85 | 3508 | 4.96 | **3445** | 1.49 | 3446 | 1.44 | -1.8 | -1.77 |
| 86 | 3588 | 1.54 | 3535 | 1.39 | **3503** | 2.87 | -1.48 | -2.37 |
| 87 | 3541 | 0.62 | 3503 | 1.64 | **3494** | 3.01 | -1.07 | -1.33 |
| 88 | 3374 | 3.48 | 3339 | 1.25 | **3334** | 2.09 | -1.04 | -1.19 |
| 89 | 3442 | 0.65 | 3382 | 1.14 | 3382 | 1.29 | -1.74 | -1.74 |
| 90 | 3839 | 1.14 | 3813 | 1.92 | **3798** | 5.44 | -0.68 | -1.07 |
| 91 | 3448 | 2.46 | 3436 | 1.57 | **3431** | 1.91 | -0.35 | -0.49 |
| 92 | 3522 | 0.81 | 3503 | 1.63 | **3502** | 3.94 | -0.54 | -0.57 |
| 93 | 3315 | 2.13 | 3294 | 1.91 | **3281** | 3.96 | -0.63 | -1.03 |
| 94 | 3486 | 1.21 | **3431** | 1.8 | 3442 | 0.9 | -1.58 | -1.26 |
| 95 | 3629 | 0.89 | 3576 | 1.61 | 3576 | 2.12 | -1.46 | -1.46 |
| 96 | 3441 | 0.63 | 3410 | 1.26 | **3409** | 1.42 | -0.9 | -0.93 |
| 97 | 3637 | 5.41 | 3623 | 2.03 | **3594** | 3.72 | -0.38 | -1.18 |
| 98 | 3311 | 2.8 | 3275 | 1.63 | **3264** | 3.51 | -1.09 | -1.42 |
| 99 | 3746 | 1.92 | 3710 | 1.2 | **3709** | 1.74 | -0.96 | -0.99 |
| 100 | 3510 | 5.33 | **3470** | 1.61 | 3474 | 1.59 | -1.14 | -1.03 |
| Avg | 3506.92 | 2.1905 | 3470.46 | 1.6046 | **3463.19** | 2.6264 | -1.0396 | -1.2465 |

Table B.9: Computational results of the proposed BLS algorithm and comparison with the best results from the four references of (Mosayebi et al., 2021) on the instances of Set IV.

| Instance | Reference algorithms | | BLS-1 | | BLS-2 | | gap-1 | gap-2 |
|---|---|---|---|---|---|---|---|---|
| | $f_{bks}$ | $t_{bks}$ | $f_{best}$ | $t_{best}$ | $f_{best}$ | $t_{best}$ | | |
| 1 | 13946 | 8.61 | 13735 | 33.85 | 13735 | 39.17 | -1.51 | -1.51 |
| 2 | 14434 | 87.97 | 14327 | 32.82 | **14264** | 49.5 | -0.74 | -1.18 |
| 3 | 13313 | 12.45 | 13192 | 32.43 | **13178** | 30.6 | -0.91 | -1.01 |
| 4 | 13682 | 25.26 | 13480 | 25.01 | 13480 | 37.93 | -1.48 | -1.48 |
| 5 | 13835 | 16.46 | 13732 | 29.15 | **13708** | 31.91 | -0.74 | -0.92 |
| 6 | 14204 | 16.16 | 14114 | 30.7 | **14087** | 41.38 | -0.63 | -0.82 |
| 7 | 13745 | 49.29 | **13619** | 25.02 | 13620 | 28.37 | -0.92 | -0.91 |
| 8 | 13831 | 13.34 | 13714 | 23.37 | **13673** | 56.85 | -0.85 | -1.14 |
| 9 | 14016 | 226.5 | 13857 | 27.91 | 13857 | 30.4 | -1.13 | -1.13 |
| 10 | 13736 | 60.12 | 13553 | 27.87 | **13543** | 31.42 | -1.33 | -1.41 |
| 11 | 13192 | 9.78 | 13080 | 25.22 | **13058** | 25.61 | -0.85 | -1.02 |
| 12 | 13563 | 14.71 | 13461 | 28.37 | **13446** | 33.03 | -0.75 | -0.86 |
| 13 | 13149 | 11.79 | 13078 | 20.3 | **13067** | 29.83 | -0.54 | -0.62 |
| 14 | 14003 | 62.32 | 13914 | 33.13 | **13900** | 50.29 | -0.64 | -0.74 |
| 15 | 13619 | 12.19 | 13568 | 21.32 | 13568 | 25.77 | -0.37 | -0.37 |
| 16 | 13309 | 43.21 | 13154 | 17.71 | **13147** | 28.34 | -1.16 | -1.22 |
| 17 | 13273 | 12.02 | 13108 | 19.61 | **13092** | 22.71 | -1.24 | -1.36 |
| 18 | 13677 | 13.04 | 13574 | 25.67 | 13574 | 28.51 | -0.75 | -0.75 |
| 19 | **13978** | 16.26 | 14009 | 30.77 | 14006 | 57.04 | 0.22 | 0.2 |
| 20 | 14095 | 14.55 | 14009 | 32.7 | **13973** | 69.3 | -0.61 | -0.87 |
| 21 | 13535 | 8.21 | 13341 | 16.36 | 13341 | 26.83 | -1.43 | -1.43 |
| 22 | 13971 | 13.5 | **13967** | 20.51 | 13974 | 29.67 | -0.03 | 0.02 |
| 23 | 13392 | 52.99 | 13224 | 31.24 | **13211** | 33.29 | -1.25 | -1.35 |
| 24 | 13991 | 16.06 | 13964 | 30.86 | 13964 | 35.25 | -0.19 | -0.19 |
| 25 | 14261 | 19.24 | 14155 | 25.19 | **14154** | 41.11 | -0.74 | -0.75 |
| 26 | 14023 | 53.14 | 14039 | 28.4 | 14015 | 35.74 | 0.11 | -0.06 |
| 27 | 13240 | 23.55 | 13161 | 18.32 | 13161 | 19.87 | -0.6 | -0.6 |
| 28 | 13698 | 12.06 | 13611 | 27.69 | **13575** | 28.87 | -0.64 | -0.9 |
| 29 | 13462 | 27.81 | **13332** | 30.25 | 13345 | 37.92 | -0.97 | -0.87 |
| 30 | 13608 | 21.15 | 13495 | 26.58 | **13486** | 27.72 | -0.83 | -0.9 |
| 31 | 13771 | 84.1 | 13583 | 27.97 | **13548** | 28.24 | -1.37 | -1.62 |
| 32 | 13745 | 15.77 | 13634 | 18.2 | **13630** | 32.29 | -0.81 | -0.84 |
| 33 | 13709 | 13.27 | **14087** | 39.4 | 14068 | 72.28 | 2.76 | 2.62 |
| 34 | 14352 | 17.8 | 14244 | 28.83 | **14206** | 40.31 | -0.75 | -1.02 |
| 35 | 12854 | 9.98 | 12667 | 26.09 | **12616** | 45.64 | -1.45 | -1.85 |
| 36 | 13869 | 7.05 | 13715 | 24.62 | **13714** | 49.3 | -1.11 | -1.12 |
| 37 | 13729 | 12.83 | 13675 | 28.29 | **13653** | 38.1 | -0.39 | -0.55 |
| 38 | 13775 | 16.77 | 13632 | 31.54 | **13628** | 35.84 | -1.04 | -1.07 |
| 39 | 14243 | 9.98 | 14147 | 28.95 | **14128** | 62.3 | -0.67 | -0.81 |
| 40 | 13962 | 17.01 | 13909 | 29.19 | **13908** | 26.05 | -0.38 | -0.39 |
| 41 | 13463 | 8.67 | 13400 | 33.55 | **13390** | 29.61 | -0.47 | -0.54 |
| 42 | 13917 | 39.85 | 13843 | 28.37 | **13810** | 41.86 | -0.53 | -0.77 |
| 43 | 13698 | 91.51 | **13521** | 26.51 | 13526 | 29.75 | -1.29 | -1.26 |
| 44 | 12963 | 10.09 | 12704 | 30.6 | **12673** | 17.93 | -2 | -2.24 |
| 45 | 13222 | 25.29 | 13087 | 23.57 | **13078** | 29.93 | -1.02 | -1.09 |
| 46 | 13883 | 14.89 | 13761 | 33.94 | **13739** | 47.84 | -0.88 | -1.04 |
| 47 | 14677 | 9.78 | 14552 | 33.99 | **14545** | 58.54 | -0.85 | -0.9 |
| 48 | 14090 | 18.41 | 14024 | 31.02 | **14016** | 38.96 | -0.47 | -0.53 |
| 49 | 14340 | 10.19 | 14145 | 27.09 | **14143** | 35.93 | -1.36 | -1.37 |
| 50 | 13896 | 8.95 | 13819 | 31.18 | **13814** | 49.7 | -0.55 | -0.59 |
| 51 | 14157 | 83.48 | 13992 | 29.98 | **13975** | 63.56 | -1.17 | -1.29 |
| 52 | 13892 | 229.9 | 13796 | 32.56 | **13759** | 62.43 | -0.69 | -0.96 |
| 53 | 13917 | 39.03 | 13786 | 33.26 | **13776** | 36.39 | -0.94 | -1.01 |
| 54 | 13731 | 39.07 | 13630 | 19.1 | **13613** | 64.17 | -0.74 | -0.86 |
| 55 | 13776 | 8.72 | 13602 | 25.21 | 13602 | 29.26 | -1.26 | -1.26 |
| 56 | 13803 | 21.43 | 13673 | 31.91 | **13666** | 35.44 | -0.94 | -0.99 |
| 57 | 13881 | 33.41 | 13802 | 25.75 | **13793** | 30.41 | -0.57 | -0.63 |
| 58 | 14371 | 40.68 | **14187** | 32.08 | 14193 | 39.89 | -1.28 | -1.24 |
| 59 | 13669 | 6.55 | 13557 | 33.5 | **13509** | 36.85 | -0.82 | -1.17 |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 60 | 13799 | 7.51 | 13678 | 26.51 | **13669** | 30.32 | -0.88 | -0.94 |
| 61 | 13584 | 23.87 | 13476 | 30.07 | **13434** | 41.71 | -0.8 | -1.1 |
| 62 | 13641 | 200.77 | 13492 | 26.85 | **13463** | 32.58 | -1.09 | -1.3 |
| 63 | 13923 | 15.61 | 13809 | 32.68 | **13795** | 37.01 | -0.82 | -0.92 |
| 64 | 13176 | 26.31 | 13053 | 21.17 | 13053 | 26.19 | -0.93 | -0.93 |
| 65 | 13719 | 8.03 | 13594 | 33.96 | **13588** | 28.63 | -0.91 | -0.95 |
| 66 | 14478 | 9.74 | 14378 | 33.51 | **14321** | 57.54 | -0.69 | -1.08 |
| 67 | 13597 | 39.4 | 13520 | 22.6 | 13520 | 24.48 | -0.57 | -0.57 |
| 68 | 13441 | 61.9 | 13345 | 21.61 | **13333** | 45.08 | -0.71 | -0.8 |
| 69 | 13667 | 29.06 | 13637 | 32.16 | **13609** | 58.51 | -0.22 | -0.42 |
| 70 | 13769 | 16.39 | 13615 | 32.27 | **13602** | 42.12 | -1.12 | -1.21 |
| 71 | 13506 | 11.19 | 13336 | 28.59 | **13289** | 62.25 | -1.26 | -1.61 |
| 72 | 13859 | 7.99 | 13701 | 21.93 | **13660** | 36.12 | -1.14 | -1.44 |
| 73 | 14165 | 41.19 | 14051 | 28.13 | **14034** | 62.26 | -0.8 | -0.92 |
| 74 | 14189 | 133.59 | 14076 | 30.08 | **14066** | 47.77 | -0.8 | -0.87 |
| 75 | 13610 | 14.23 | 13565 | 31.31 | **13537** | 40.55 | -0.33 | -0.54 |
| 76 | 13982 | 15.76 | 13813 | 23.1 | 13813 | 41.84 | -1.21 | -1.21 |
| 77 | 13331 | 5.86 | 13139 | 22.44 | **13129** | 38.59 | -1.44 | -1.52 |
| 78 | 14157 | 100.31 | 14063 | 23.25 | **14008** | 42.93 | -0.66 | -1.05 |
| 79 | 13644 | 8.09 | 13495 | 26.62 | **13474** | 28.82 | -1.09 | -1.25 |
| 80 | 13606 | 29.39 | **13487** | 32.02 | 13511 | 25.87 | -0.87 | -0.7 |
| 81 | 13109 | 11.71 | 13003 | 18.06 | **12949** | 44.17 | -0.81 | -1.22 |
| 82 | 13979 | 15.17 | 13865 | 31.62 | **13833** | 55.26 | -0.82 | -1.04 |
| 83 | 13240 | 19.54 | 13036 | 26.31 | **13015** | 31.86 | -1.54 | -1.7 |
| 84 | 13710 | 23.06 | 13558 | 32.88 | **13548** | 25.89 | -1.11 | -1.18 |
| 85 | 13194 | 6.26 | 13123 | 26.36 | **13095** | 42.5 | -0.54 | -0.75 |
| 86 | 14443 | 8.14 | 14335 | 23.56 | **14295** | 69.77 | -0.75 | -1.02 |
| 87 | 13115 | 28.09 | **13022** | 33.97 | 13041 | 22.35 | -0.71 | -0.56 |
| 88 | 13559 | 6.87 | 13432 | 19.51 | **13404** | 28.86 | -0.94 | -1.14 |
| 89 | 13460 | 26.1 | 13330 | 25.61 | 13330 | 23.65 | -0.97 | -0.97 |
| 90 | 14117 | 12.97 | 13984 | 33.43 | **13980** | 57.62 | -0.94 | -0.97 |
| 91 | 14093 | 28.81 | 13997 | 30.54 | **13990** | 45.51 | -0.68 | -0.73 |
| 92 | 13648 | 15.04 | 13481 | 21.65 | **13472** | 37.91 | -1.22 | -1.29 |
| 93 | 13764 | 13.44 | 13662 | 28.79 | **13658** | 66.2 | -0.74 | -0.77 |
| 94 | 13889 | 13.85 | 13769 | 31.39 | **13739** | 54.22 | -0.86 | -1.08 |
| 95 | 13888 | 13.4 | **13850** | 31.45 | 13855 | 27.41 | -0.27 | -0.24 |
| 96 | 13540 | 15.88 | 13366 | 23.18 | **13331** | 23.35 | -1.29 | -1.54 |
| 97 | 13180 | 128.84 | 13111 | 25.86 | **13097** | 69.16 | -0.52 | -0.63 |
| 98 | 13606 | 101.99 | 13357 | 32.85 | **13349** | 58.25 | -1.83 | -1.89 |
| 99 | 14919 | 18.12 | 14758 | 31.25 | **14725** | 65.41 | -1.08 | -1.3 |
| 100 | 13236 | 161.45 | 13092 | 21.33 | **13091** | 27.79 | -1.09 | -1.1 |
| Avg | 13756.68 | 33.9312 | 13642.04 | 27.75 | **13627.03** | 39.8 | -0.8352 | -0.9443 |

Table B.10: Results of the four reimplemented algorithms of (Mosayebi et al., 2021) on the instances of Set II.

| Instance | Pro.I | | Pro.II | | Pro.III | | Pro.IV | |
|---|---|---|---|---|---|---|---|---|
| | $f_{best}$ | $t_{best}$ | $f_{best}$ | $t_{best}$ | $f_{best}$ | $t_{best}$ | $f_{best}$ | $t_{best}$ |
| 1 | 319 | 0.00114 | 320 | 0.00062 | 319 | 0.00114 | 319 | 0.0012 |
| 2 | 280 | 0.00085 | 275 | 0.00061 | 280 | 0.00086 | 273 | 0.00116 |
| 3 | 296 | 0.00188 | 288 | 0.00117 | 296 | 0.00187 | 292 | 0.00231 |
| 4 | 289 | 0.00189 | 293 | 0.00106 | 289 | 0.00153 | 289 | 0.00176 |
| 5 | 282 | 0.00247 | 282 | 0.00122 | 282 | 0.0018 | 282 | 0.00262 |
| 6 | 309 | 0.00127 | 305 | 0.00063 | 309 | 0.00102 | 316 | 0.0013 |
| 7 | 288 | 0.00145 | 292 | 0.00081 | 288 | 0.00121 | 286 | 0.00254 |
| 8 | 264 | 0.001 | 254 | 0.00071 | 264 | 0.00089 | 264 | 0.00158 |
| 9 | 293 | 0.00081 | 290 | 0.00066 | 293 | 0.00082 | 298 | 0.00169 |
| 10 | 316 | 0.0015 | 320 | 0.00092 | 316 | 0.0016 | 313 | 0.00167 |
| 11 | 305 | 0.00149 | 299 | 0.00079 | 305 | 0.00161 | 306 | 0.00172 |
| 12 | 284 | 0.00322 | 287 | 0.00119 | 284 | 0.00294 | 284 | 0.00255 |
| 13 | 276 | 0.00173 | 268 | 0.00127 | 276 | 0.00175 | 272 | 0.00174 |
| 14 | 270 | 0.00107 | 270 | 0.0008 | 270 | 0.00106 | 270 | 0.00111 |
| 15 | 277 | 0.00092 | 271 | 0.00056 | 277 | 0.00089 | 273 | 0.00147 |
| 16 | 303 | 0.00237 | 299 | 0.00125 | 303 | 0.00196 | 303 | 0.0019 |
| 17 | 271 | 0.00112 | 265 | 0.001 | 271 | 0.00098 | 271 | 0.0012 |
| 18 | 258 | 0.00082 | 258 | 0.0008 | 258 | 0.00071 | 258 | 0.00098 |
| 19 | 251 | 0.00095 | 248 | 0.00076 | 251 | 0.00077 | 257 | 0.00123 |
| 20 | 293 | 0.00187 | 289 | 0.0006 | 293 | 0.00148 | 295 | 0.00216 |
| 21 | 291 | 0.00114 | 285 | 0.00089 | 291 | 0.001 | 291 | 0.00137 |
| 22 | 236 | 0.00078 | 239 | 0.00064 | 236 | 0.00075 | 236 | 0.00079 |
| 23 | 288 | 0.0013 | 279 | 0.0007 | 288 | 0.00107 | 288 | 0.00116 |
| 24 | 299 | 0.00204 | 298 | 0.00099 | 299 | 0.00208 | 299 | 0.00205 |
| 25 | 256 | 0.0014 | 240 | 0.00074 | 256 | 0.00136 | 256 | 0.00123 |
| 26 | 305 | 0.00092 | 291 | 0.00076 | 305 | 0.00088 | 305 | 0.00077 |
| 27 | 297 | 0.00184 | 305 | 0.00115 | 297 | 0.00178 | 297 | 0.00188 |
| 28 | 285 | 0.00191 | 288 | 0.00095 | 285 | 0.00178 | 290 | 0.00139 |
| 29 | 299 | 0.00232 | 299 | 0.00112 | 299 | 0.00238 | 304 | 0.00155 |
| 30 | 307 | 0.00181 | 305 | 0.00101 | 307 | 0.00183 | 304 | 0.00205 |
| 31 | 268 | 0.00115 | 261 | 0.00083 | 267 | 0.00108 | 267 | 0.00112 |
| 32 | 332 | 0.00123 | 328 | 0.0008 | 332 | 0.0011 | 325 | 0.00099 |
| 33 | 312 | 0.00165 | 309 | 0.00146 | 306 | 0.00181 | 312 | 0.00172 |
| 34 | 294 | 0.00154 | 311 | 0.00096 | 294 | 0.00151 | 294 | 0.00161 |
| 35 | 283 | 0.00184 | 290 | 0.00122 | 283 | 0.00186 | 297 | 0.00156 |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 36 | 263 | 0.0016 | 265 | 0.00081 | 263 | 0.00148 | 263 | 0.00156 |
| 37 | 284 | 0.0011 | 279 | 0.00082 | 284 | 0.00139 | 280 | 0.00173 |
| 38 | 251 | 0.00106 | 249 | 0.00064 | 251 | 0.0009 | 251 | 0.00083 |
| 39 | 280 | 0.00138 | 276 | 0.00086 | 280 | 0.0012 | 280 | 0.00141 |
| 40 | 261 | 0.00287 | 256 | 0.00116 | 261 | 0.00229 | 261 | 0.00248 |
| 41 | 295 | 0.00088 | 287 | 0.00052 | 295 | 0.00087 | 299 | 0.00148 |
| 42 | 302 | 0.00101 | 300 | 0.0008 | 302 | 0.00091 | 297 | 0.00122 |
| 43 | 298 | 0.00226 | 297 | 0.00129 | 299 | 0.00183 | 299 | 0.00244 |
| 44 | 272 | 0.00142 | 277 | 0.00084 | 271 | 0.0011 | 271 | 0.00122 |
| 45 | 337 | 0.0015 | 325 | 0.00163 | 337 | 0.00149 | 342 | 0.00231 |
| 46 | 252 | 0.00088 | 248 | 0.00066 | 252 | 0.00081 | 252 | 0.00091 |
| 47 | 330 | 0.0018 | 325 | 0.00086 | 330 | 0.00113 | 330 | 0.00132 |
| 48 | 300 | 0.00145 | 302 | 0.00109 | 300 | 0.00146 | 307 | 0.00178 |
| 49 | 269 | 0.00174 | 265 | 0.00105 | 280 | 0.00156 | 276 | 0.00117 |
| 50 | 293 | 0.00195 | 294 | 0.0007 | 293 | 0.00191 | 293 | 0.00147 |
| 51 | 278 | 0.00097 | 282 | 0.00072 | 275 | 0.00111 | 275 | 0.00096 |
| 52 | 341 | 0.00155 | 351 | 0.00088 | 341 | 0.00155 | 341 | 0.00175 |
| 53 | 288 | 0.00155 | 286 | 0.00124 | 288 | 0.0017 | 294 | 0.00161 |
| 54 | 273 | 0.00077 | 267 | 0.00071 | 273 | 0.00077 | 273 | 0.00084 |
| 55 | 307 | 0.00221 | 304 | 0.0005 | 307 | 0.00193 | 307 | 0.0018 |
| 56 | 289 | 0.00084 | 289 | 0.00051 | 289 | 0.00086 | 311 | 0.00107 |
| 57 | 257 | 0.00154 | 260 | 0.00107 | 257 | 0.00154 | 260 | 0.00168 |
| 58 | 341 | 0.00134 | 347 | 0.00092 | 341 | 0.00132 | 341 | 0.00154 |
| 59 | 293 | 0.00211 | 288 | 0.00107 | 293 | 0.0019 | 293 | 0.00217 |
| 60 | 259 | 0.00107 | 257 | 0.001 | 259 | 0.00103 | 261 | 0.00144 |
| 61 | 307 | 0.00187 | 309 | 0.00072 | 307 | 0.00183 | 318 | 0.0021 |
| 62 | 289 | 0.00162 | 309 | 0.00098 | 289 | 0.00156 | 289 | 0.00172 |
| 63 | 275 | 0.00151 | 286 | 0.00075 | 275 | 0.00168 | 274 | 0.00145 |
| 64 | 297 | 0.00175 | 290 | 0.00082 | 297 | 0.00179 | 311 | 0.00126 |
| 65 | 272 | 0.00107 | 275 | 0.00058 | 272 | 0.00086 | 269 | 0.00112 |
| 66 | 294 | 0.00164 | 296 | 0.00107 | 294 | 0.00135 | 290 | 0.00188 |
| 67 | 325 | 0.00157 | 320 | 0.00121 | 325 | 0.00131 | 325 | 0.00172 |
| 68 | 301 | 0.00179 | 302 | 0.00085 | 301 | 0.00152 | 302 | 0.00191 |
| 69 | 309 | 0.00188 | 307 | 0.00097 | 309 | 0.00162 | 300 | 0.00226 |
| 70 | 273 | 0.00123 | 280 | 0.00078 | 273 | 0.00091 | 291 | 0.00109 |
| 71 | 295 | 0.00123 | 298 | 0.00083 | 295 | 0.00111 | 301 | 0.00117 |
| 72 | 259 | 0.00091 | 259 | 0.0008 | 259 | 0.00082 | 259 | 0.00096 |
| 73 | 278 | 0.00177 | 281 | 0.00092 | 278 | 0.0015 | 281 | 0.00218 |
| 74 | 271 | 0.00088 | 277 | 0.00101 | 271 | 0.00119 | 271 | 0.0014 |
| 75 | 216 | 0.0007 | 212 | 0.00063 | 212 | 0.00063 | 212 | 0.00089 |
| 76 | 300 | 0.00101 | 297 | 0.00084 | 300 | 0.001 | 300 | 0.00106 |
| 77 | 275 | 0.00174 | 281 | 0.00064 | 275 | 0.00165 | 274 | 0.00109 |
| 78 | 272 | 0.0014 | 258 | 0.00099 | 272 | 0.0014 | 263 | 0.00195 |
| 79 | 281 | 0.0013 | 282 | 0.00093 | 281 | 0.00133 | 286 | 0.00233 |
| 80 | 283 | 0.00122 | 280 | 0.00079 | 283 | 0.00134 | 281 | 0.00152 |
| 81 | 273 | 0.00113 | 271 | 0.00075 | 273 | 0.0011 | 259 | 0.00148 |
| 82 | 361 | 0.00153 | 347 | 0.001 | 361 | 0.00147 | 345 | 0.00187 |
| 83 | 259 | 0.00175 | 261 | 0.00085 | 259 | 0.00168 | 259 | 0.00171 |
| 84 | 274 | 0.0009 | 275 | 0.00071 | 274 | 0.00084 | 272 | 0.00108 |
| 85 | 300 | 0.00124 | 306 | 0.00095 | 300 | 0.00117 | 302 | 0.00125 |
| 86 | 341 | 0.00115 | 338 | 0.00063 | 341 | 0.0012 | 343 | 0.0014 |
| 87 | 275 | 0.00094 | 261 | 0.00087 | 275 | 0.00092 | 270 | 0.00116 |
| 88 | 312 | 0.00096 | 311 | 0.00101 | 312 | 0.00097 | 305 | 0.00169 |
| 89 | 293 | 0.00112 | 288 | 0.00084 | 293 | 0.00114 | 297 | 0.00123 |
| 90 | 329 | 0.00127 | 322 | 0.00108 | 329 | 0.00134 | 324 | 0.00325 |
| 91 | 290 | 0.00183 | 281 | 0.00115 | 290 | 0.00158 | 290 | 0.00129 |
| 92 | 323 | 0.00147 | 300 | 0.00061 | 323 | 0.00145 | 319 | 0.00142 |
| 93 | 318 | 0.00228 | 309 | 0.0012 | 318 | 0.00189 | 318 | 0.00159 |
| 94 | 311 | 0.00175 | 309 | 0.00105 | 311 | 0.00152 | 312 | 0.00152 |
| 95 | 262 | 0.00177 | 265 | 0.00094 | 260 | 0.00126 | 260 | 0.00128 |
| 96 | 280 | 0.00128 | 268 | 0.00092 | 280 | 0.0011 | 277 | 0.00202 |
| 97 | 256 | 0.00083 | 266 | 0.00053 | 256 | 0.00106 | 256 | 0.00128 |
| 98 | 320 | 0.00113 | 298 | 0.00068 | 320 | 0.00093 | 314 | 0.00139 |
| 99 | 287 | 0.00192 | 278 | 0.00135 | 287 | 0.0017 | 281 | 0.00205 |
| 100 | 258 | 0.00149 | 261 | 0.00074 | 249 | 0.00083 | 249 | 0.00074 |

Table B.11: Results of the four reimplemented algorithms of (Mosayebi et al., 2021) on the instances of Set III.

| Instance | Pro.I | | Pro.II | | Pro.III | | Pro.IV | |
|---|---|---|---|---|---|---|---|---|
| | $f_{best}$ | $t_{best}$ | $f_{best}$ | $t_{best}$ | $f_{best}$ | $t_{best}$ | $f_{best}$ | $t_{best}$ |
| 1 | 3253 | 5.22 | 3235 | 0.71 | 3275 | 3.14 | 3257 | 2.19 |
| 2 | 3525 | 2.66 | 3494 | 0.57 | 3525 | 1.88 | 3525 | 1.69 |
| 3 | 3467 | 3.59 | 3477 | 0.8 | 3485 | 1.83 | 3481 | 2.23 |
| 4 | 3480 | 1.39 | 3486 | 0.68 | 3480 | 1.37 | 3465 | 4.44 |
| 5 | 3726 | 2.82 | 3673 | 0.66 | 3726 | 1.84 | 3700 | 3.21 |
| 6 | 3289 | 2.16 | 3273 | 0.61 | 3289 | 1.93 | 3284 | 1.59 |
| 7 | 3614 | 10.17 | 3571 | 0.93 | 3601 | 8 | 3601 | 6.79 |
| 8 | 3638 | 3.59 | 3604 | 0.68 | 3638 | 2.41 | 3635 | 6.16 |
| 9 | 3372 | 3.25 | 3335 | 0.76 | 3372 | 2.3 | 3373 | 1.77 |
| 10 | 3588 | 9.2 | 3544 | 0.89 | 3588 | 8.57 | 3551 | 3.98 |
| 11 | 3421 | 4.38 | 3401 | 0.73 | 3421 | 4 | 3421 | 3.86 |
| 12 | 3510 | 1.88 | 3521 | 0.43 | 3510 | 1.78 | 3555 | 3.23 |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 13 | 3586 | 3.69 | 3523 | 1.14 | 3586 | 3.18 | 3550 | 2.86 |
| 14 | 3498 | 2.21 | 3480 | 0.42 | 3494 | 2.04 | 3494 | 1.79 |
| 15 | 3472 | 3.38 | 3484 | 0.85 | 3462 | 2.04 | 3493 | 2.5 |
| 16 | 3699 | 7.22 | 3725 | 1 | 3699 | 6.72 | 3723 | 6.76 |
| 17 | 3579 | 6.45 | 3588 | 0.76 | 3579 | 4.62 | 3571 | 7.89 |
| 18 | 3619 | 2.08 | 3593 | 0.99 | 3619 | 1.91 | 3658 | 3.64 |
| 19 | 3534 | 3.48 | 3530 | 0.62 | 3534 | 2.48 | 3501 | 3.68 |
| 20 | 3510 | 2.16 | 3504 | 0.72 | 3510 | 1.51 | 3524 | 7.37 |
| 21 | 3611 | 5.03 | 3589 | 0.5 | 3590 | 5.93 | 3628 | 4.1 |
| 22 | 3695 | 7.47 | 3714 | 1.15 | 3695 | 6.96 | 3685 | 9.75 |
| 23 | 3412 | 1.66 | 3416 | 0.38 | 3398 | 2.66 | 3401 | 1.35 |
| 24 | 3476 | 2.41 | 3481 | 0.99 | 3476 | 1.53 | 3491 | 1.95 |
| 25 | 3671 | 7.15 | 3676 | 0.99 | 3671 | 6.97 | 3671 | 4.75 |
| 26 | 3551 | 3.3 | 3477 | 0.73 | 3551 | 3.13 | 3551 | 2.81 |
| 27 | 3601 | 1.91 | 3576 | 0.9 | 3601 | 1.95 | 3606 | 5 |
| 28 | 3562 | 1.59 | 3528 | 0.73 | 3562 | 1.52 | 3505 | 0.82 |
| 29 | 3365 | 1.85 | 3327 | 0.53 | 3365 | 1.74 | 3376 | 2.66 |
| 30 | 3255 | 1.69 | 3276 | 0.67 | 3255 | 1.6 | 3256 | 1.48 |
| 31 | 3335 | 2.82 | 3324 | 0.37 | 3335 | 2.02 | 3346 | 2.98 |
| 32 | 3605 | 2.27 | 3559 | 0.91 | 3605 | 1.51 | 3608 | 2.91 |
| 33 | 3703 | 3.15 | 3667 | 0.96 | 3703 | 2.24 | 3681 | 1.51 |
| 34 | 3523 | 4.95 | 3502 | 0.48 | 3492 | 5.32 | 3492 | 4.05 |
| 35 | 3507 | 1.96 | 3509 | 0.48 | 3507 | 1.36 | 3505 | 1.87 |
| 36 | 3523 | 2.07 | 3467 | 0.4 | 3523 | 1.39 | 3532 | 3.06 |
| 37 | 3663 | 4.94 | 3668 | 0.98 | 3663 | 4.61 | 3679 | 8.39 |
| 38 | 3643 | 4.05 | 3621 | 0.91 | 3643 | 3.84 | 3660 | 6.25 |
| 39 | 3435 | 3.84 | 3432 | 0.53 | 3435 | 3.58 | 3435 | 3.24 |
| 40 | 3297 | 2.51 | 3293 | 0.35 | 3309 | 2.02 | 3352 | 2.41 |
| 41 | 3473 | 2.13 | 3465 | 0.8 | 3473 | 2.07 | 3486 | 2.5 |
| 42 | 3527 | 2.65 | 3506 | 0.79 | 3527 | 2.39 | 3516 | 7.2 |
| 43 | 3434 | 1.97 | 3415 | 0.71 | 3434 | 1.95 | 3406 | 1.68 |
| 44 | 3616 | 6.17 | 3634 | 0.82 | 3616 | 5.87 | 3648 | 4.77 |
| 45 | 3599 | 1.76 | 3612 | 0.8 | 3599 | 2.26 | 3587 | 2.61 |
| 46 | 3431 | 1.44 | 3388 | 0.61 | 3431 | 0.93 | 3386 | 2.27 |
| 47 | 3373 | 1.64 | 3335 | 0.69 | 3373 | 1.11 | 3342 | 4.5 |
| 48 | 3620 | 2.01 | 3558 | 0.78 | 3617 | 1.78 | 3605 | 2.07 |
| 49 | 3613 | 4.83 | 3602 | 0.83 | 3613 | 3.42 | 3613 | 3.19 |
| 50 | 3608 | 4.39 | 3601 | 0.54 | 3608 | 4.25 | 3594 | 8.24 |
| 51 | 3409 | 2.15 | 3373 | 0.52 | 3409 | 2.05 | 3409 | 1.72 |
| 52 | 3346 | 2.53 | 3317 | 0.67 | 3346 | 2.2 | 3328 | 0.94 |
| 53 | 3543 | 4.12 | 3528 | 0.44 | 3543 | 3.79 | 3538 | 3.55 |
| 54 | 3368 | 7.44 | 3382 | 0.75 | 3368 | 6.64 | 3375 | 5.09 |
| 55 | 3801 | 7.3 | 3739 | 0.57 | 3801 | 7.39 | 3801 | 5.69 |
| 56 | 3709 | 2.68 | 3682 | 0.51 | 3709 | 2.5 | 3749 | 2.47 |
| 57 | 3709 | 8.86 | 3748 | 1.08 | 3709 | 7.97 | 3709 | 7.71 |
| 58 | 3658 | 3.77 | 3633 | 0.63 | 3650 | 3.36 | 3661 | 4.94 |
| 59 | 3338 | 4.43 | 3302 | 0.4 | 3330 | 2.93 | 3338 | 1.96 |
| 60 | 3392 | 3.37 | 3404 | 0.42 | 3392 | 2.53 | 3405 | 2.13 |
| 61 | 3422 | 5.27 | 3422 | 0.47 | 3421 | 3.92 | 3435 | 2.66 |
| 62 | 3641 | 2.59 | 3609 | 0.74 | 3630 | 3.7 | 3625 | 3.68 |
| 63 | 3570 | 8.53 | 3558 | 0.51 | 3570 | 8.27 | 3583 | 7.63 |
| 64 | 3470 | 2.03 | 3470 | 0.48 | 3470 | 1.93 | 3472 | 2.92 |
| 65 | 3758 | 1.82 | 3732 | 1.01 | 3758 | 3.26 | 3758 | 2.95 |
| 66 | 3650 | 4.22 | 3661 | 0.78 | 3645 | 4.25 | 3660 | 3.3 |
| 67 | 3427 | 2.38 | 3460 | 0.65 | 3427 | 2.38 | 3427 | 2.18 |
| 68 | 3343 | 2.11 | 3375 | 0.8 | 3350 | 1.37 | 3350 | 1.14 |
| 69 | 3658 | 5.09 | 3641 | 0.56 | 3658 | 3.65 | 3658 | 3.01 |
| 70 | 3532 | 4.2 | 3520 | 0.89 | 3532 | 3.94 | 3532 | 3.38 |
| 71 | 3600 | 5.06 | 3599 | 0.76 | 3600 | 4.75 | 3633 | 4.53 |
| 72 | 3627 | 10.32 | 3630 | 1.06 | 3627 | 9.01 | 3623 | 5.24 |
| 73 | 3413 | 4.51 | 3428 | 0.4 | 3413 | 4.16 | 3413 | 3.35 |
| 74 | 3330 | 4.14 | 3281 | 0.6 | 3330 | 3.79 | 3330 | 3.34 |
| 75 | 3390 | 3.65 | 3416 | 0.69 | 3390 | 2.66 | 3424 | 1.88 |
| 76 | 3662 | 3.63 | 3645 | 0.85 | 3662 | 2.57 | 3657 | 4.65 |
| 77 | 3495 | 3.23 | 3472 | 0.56 | 3495 | 3.04 | 3478 | 2.98 |
| 78 | 3490 | 4.22 | 3480 | 0.51 | 3490 | 3.12 | 3444 | 3.48 |
| 79 | 3695 | 1.55 | 3676 | 1.07 | 3695 | 2.78 | 3680 | 2.8 |
| 80 | 3609 | 2.34 | 3554 | 0.76 | 3609 | 2.13 | 3561 | 3.18 |
| 81 | 3456 | 1.51 | 3446 | 0.32 | 3456 | 1.48 | 3437 | 3.66 |
| 82 | 3475 | 1.92 | 3482 | 0.77 | 3475 | 1.87 | 3475 | 1.84 |
| 83 | 3433 | 2.52 | 3428 | 0.73 | 3433 | 2.36 | 3443 | 3.66 |
| 84 | 3567 | 4.36 | 3588 | 0.47 | 3567 | 3.18 | 3570 | 8.63 |
| 85 | 3514 | 3.77 | 3541 | 0.77 | 3514 | 2.71 | 3508 | 4.96 |
| 86 | 3588 | 1.54 | 3595 | 0.36 | 3588 | 0.97 | 3611 | 3.14 |
| 87 | 3589 | 4.27 | 3541 | 0.62 | 3589 | 3.08 | 3554 | 9.34 |
| 88 | 3374 | 3.48 | 3386 | 0.66 | 3374 | 3.45 | 3396 | 4.76 |
| 89 | 3451 | 2.27 | 3442 | 0.65 | 3451 | 2.14 | 3459 | 1.92 |
| 90 | 3876 | 3.57 | 3839 | 1.14 | 3876 | 3.7 | 3895 | 2.34 |
| 91 | 3470 | 1.7 | 3502 | 0.6 | 3470 | 2.22 | 3448 | 2.46 |
| 92 | 3559 | 3.64 | 3522 | 0.81 | 3559 | 2.68 | 3531 | 4.55 |
| 93 | 3316 | 3.9 | 3319 | 0.58 | 3315 | 2.13 | 3315 | 2.53 |
| 94 | 3486 | 1.21 | 3495 | 0.62 | 3486 | 0.78 | 3493 | 3.34 |
| 95 | 3645 | 4.54 | 3629 | 0.89 | 3645 | 4.22 | 3645 | 4.34 |
| 96 | 3458 | 1.65 | 3441 | 0.63 | 3458 | 1.55 | 3458 | 1.39 |
| 97 | 3676 | 3.34 | 3646 | 1.04 | 3676 | 3.2 | 3637 | 5.41 |
| 98 | 3311 | 2.8 | 3325 | 0.7 | 3311 | 2.61 | 3354 | 2.25 |
| 99 | 3752 | 5.11 | 3752 | 0.87 | 3752 | 8.32 | 3746 | 1.92 |
| 100 | 3510 | 5.33 | 3526 | 0.97 | 3510 | 5.12 | 3523 | 4.01 |

660

Table B.12: The results of the four reimplemented algorithms of (Mosayebi et al., 2021) on the instances of Set IV.

| Instance | Pro.I | | Pro.II | | Pro.III | | Pro.IV | |
|---|---|---|---|---|---|---|---|---|
| | $f_{best}$ | $t_{best}$ | $f_{best}$ | $t_{best}$ | $f_{best}$ | $t_{best}$ | $f_{best}$ | $t_{best}$ |
| 1 | 13980 | 112.97 | 13946 | 8.61 | 13982 | 119.2 | 13982 | 110.52 |
| 2 | 14542 | 186.77 | 14467 | 18.05 | 14434 | 87.97 | 14434 | 78.37 |
| 3 | 13441 | 61.15 | 13313 | 12.45 | 13450 | 68.74 | 13366 | 46.72 |
| 4 | 13696 | 61.44 | 13690 | 13.8 | 13690 | 80.7 | 13682 | 115.7 |
| 5 | 13901 | 109.25 | 13835 | 16.46 | 13901 | 111.6 | 13937 | 168.09 |
| 6 | 14286 | 142.6 | 14204 | 16.16 | 14286 | 88.96 | 14250 | 29.02 |
| 7 | 13745 | 49.29 | 13753 | 13.17 | 13745 | 35.94 | 13745 | 30.62 |
| 8 | 13855 | 180.92 | 13831 | 13.34 | 13855 | 137.84 | 13858 | 130.55 |
| 9 | 14134 | 143.34 | 14097 | 15.98 | 14016 | 226.5 | 14117 | 161.14 |
| 10 | 13736 | 60.12 | 13754 | 12.06 | 13736 | 43.62 | 13762 | 117.78 |
| 11 | 13282 | 42.88 | 13192 | 9.78 | 13282 | 40.61 | 13296 | 18.58 |
| 12 | 13610 | 177.31 | 13563 | 14.71 | 13698 | 115.34 | 13647 | 91.19 |
| 13 | 13257 | 69.66 | 13149 | 11.79 | 13257 | 62.79 | 13193 | 84.14 |
| 14 | 14003 | 62.32 | 14034 | 15.42 | 14003 | 72.62 | 14013 | 143.03 |
| 15 | 13730 | 50.69 | 13619 | 12.19 | 13730 | 46.23 | 13709 | 55.82 |
| 16 | 13309 | 43.21 | 13356 | 10.43 | 13309 | 38.51 | 13372 | 38.96 |
| 17 | 13310 | 26.48 | 13273 | 12.02 | 13323 | 20.85 | 13296 | 54.96 |
| 18 | 13761 | 55.35 | 13677 | 13.04 | 13761 | 49.35 | 13857 | 116.39 |
| 19 | 14143 | 145.96 | 13978 | 16.26 | 14108 | 53.59 | 14123 | 126.2 |
| 20 | 14134 | 88.64 | 14095 | 14.55 | 14134 | 86.17 | 14152 | 69.08 |
| 21 | 13566 | 48.7 | 13535 | 8.21 | 13566 | 47.47 | 13569 | 184.87 |
| 22 | 14141 | 61.07 | 13971 | 13.5 | 14141 | 60.34 | 14137 | 118.61 |
| 23 | 13392 | 52.99 | 13393 | 12.5 | 13392 | 49.91 | 13468 | 55.71 |
| 24 | 14161 | 136.83 | 13991 | 16.06 | 14142 | 119.77 | 14108 | 155.87 |
| 25 | 14381 | 74.53 | 14261 | 19.24 | 14379 | 71.52 | 14360 | 51.34 |
| 26 | 14206 | 67.36 | 14068 | 8.44 | 14206 | 66.57 | 14023 | 350.21 |
| 27 | 13245 | 58.94 | 13296 | 9.47 | 13245 | 54.93 | 13240 | 60.4 |
| 28 | 13757 | 57.58 | 13698 | 12.06 | 13707 | 43.05 | 13709 | 47.04 |
| 29 | 13528 | 86.76 | 13535 | 12.45 | 13528 | 85.3 | 13462 | 100.85 |
| 30 | 13759 | 93.67 | 13656 | 12.01 | 13759 | 90.48 | 13608 | 67.34 |
| 31 | 13771 | 84.1 | 13799 | 6.32 | 13771 | 81.2 | 13792 | 122.84 |
| 32 | 13889 | 157.23 | 13745 | 15.77 | 13873 | 150.2 | 13859 | 137.1 |
| 33 | 13843 | 103.04 | 13709 | 13.27 | 13843 | 98.4 | 13843 | 86.08 |
| 34 | 14404 | 36.25 | 14352 | 17.8 | 14404 | 35.48 | 14403 | 186.12 |
| 35 | 12885 | 146.16 | 12854 | 9.98 | 12885 | 146.65 | 12912 | 124.49 |
| 36 | 13940 | 32.36 | 13869 | 7.05 | 13940 | 30.06 | 13906 | 169.63 |
| 37 | 13862 | 64.51 | 13729 | 12.83 | 13862 | 63.53 | 13895 | 67.03 |
| 38 | 13850 | 82.97 | 13775 | 16.77 | 13850 | 80.65 | 13837 | 124.29 |
| 39 | 14324 | 199.25 | 14243 | 9.98 | 14324 | 191.65 | 14324 | 160.37 |
| 40 | 14038 | 45.76 | 13962 | 17.01 | 14038 | 43.8 | 13979 | 75.72 |
| 41 | 13629 | 144.78 | 13463 | 8.67 | 13629 | 135.82 | 13628 | 122.49 |
| 42 | 13974 | 63.52 | 13947 | 14.55 | 13974 | 61.07 | 13917 | 207.2 |
| 43 | 13698 | 91.51 | 13724 | 12.66 | 13698 | 87.37 | 13768 | 35.62 |
| 44 | 12991 | 103.86 | 12963 | 10.09 | 12987 | 97.94 | 13076 | 22.98 |
| 45 | 13253 | 123.3 | 13302 | 10.49 | 13253 | 115.81 | 13222 | 95.02 |
| 46 | 13950 | 99.24 | 13883 | 14.89 | 13950 | 94.75 | 13950 | 84.63 |
| 47 | 14748 | 43.61 | 14677 | 9.78 | 14748 | 44.4 | 14709 | 69.29 |
| 48 | 14142 | 66.63 | 14090 | 18.41 | 14142 | 66.19 | 14162 | 95.22 |
| 49 | 14341 | 75.18 | 14340 | 10.19 | 14341 | 71.03 | 14418 | 62.71 |
| 50 | 13987 | 89.82 | 13896 | 8.95 | 13987 | 84.89 | 13945 | 47.7 |
| 51 | 14157 | 83.48 | 14158 | 16.57 | 14157 | 81.57 | 14198 | 63.87 |
| 52 | 13892 | 229.9 | 13956 | 16.96 | 13892 | 217.55 | 14004 | 235.72 |
| 53 | 13925 | 243.13 | 13992 | 16.42 | 13925 | 216.86 | 13917 | 88.71 |
| 54 | 13753 | 193.36 | 13879 | 12.8 | 13753 | 178.52 | 13731 | 151.36 |
| 55 | 13840 | 59.93 | 13776 | 8.72 | 13840 | 55.43 | 13842 | 54.29 |
| 56 | 13846 | 129.6 | 13840 | 6.61 | 13846 | 95.87 | 13803 | 160.76 |
| 57 | 14015 | 35.61 | 13952 | 15.15 | 14015 | 32.6 | 13881 | 124.94 |
| 58 | 14550 | 128.35 | 14441 | 15.89 | 14550 | 116.8 | 14371 | 128.47 |
| 59 | 13742 | 54.31 | 13669 | 6.55 | 13742 | 50.29 | 13825 | 73.69 |
| 60 | 13835 | 88.57 | 13799 | 7.51 | 13809 | 134.73 | 13893 | 109.28 |
| 61 | 13680 | 55.56 | 13584 | 11.78 | 13680 | 57.03 | 13613 | 29.1 |
| 62 | 13641 | 200.77 | 13656 | 12.02 | 13707 | 127.78 | 13678 | 122.81 |
| 63 | 13948 | 47.7 | 13923 | 15.61 | 13948 | 45.79 | 14025 | 132.62 |
| 64 | 13299 | 81.65 | 13190 | 11.63 | 13299 | 60.08 | 13176 | 89.47 |
| 65 | 13773 | 56.59 | 13719 | 8.03 | 13773 | 54.44 | 13824 | 134.28 |
| 66 | 14553 | 326.79 | 14478 | 9.74 | 14553 | 313.5 | 14553 | 277.38 |
| 67 | 13617 | 37.2 | 13642 | 7.58 | 13597 | 39.4 | 13639 | 123.22 |
| 68 | 13530 | 32.35 | 13468 | 5.89 | 13441 | 61.9 | 13494 | 80.73 |
| 69 | 13702 | 70.72 | 13792 | 13.92 | 13702 | 67.02 | 13667 | 41.69 |
| 70 | 13826 | 88.44 | 13769 | 16.39 | 13826 | 81.64 | 13800 | 84.63 |
| 71 | 13565 | 197.43 | 13506 | 11.19 | 13565 | 190.37 | 13579 | 180.11 |
| 72 | 13888 | 80.25 | 13859 | 7.99 | 13873 | 78.43 | 13896 | 86.59 |
| 73 | 14343 | 98.04 | 14200 | 13.56 | 14343 | 91.99 | 14165 | 79.8 |
| 74 | 14202 | 182.64 | 14235 | 14.4 | 14189 | 133.59 | 14259 | 121.24 |
| 75 | 13792 | 46.84 | 13610 | 14.23 | 13792 | 44.38 | 13792 | 95.29 |
| 76 | 14063 | 149.41 | 13982 | 15.76 | 14063 | 138.4 | 14063 | 121.26 |
| 77 | 13386 | 29.01 | 13331 | 5.86 | 13386 | 26.89 | 13467 | 59.7 |
| 78 | 14263 | 120.3 | 14184 | 9.02 | 14157 | 100.31 | 14160 | 92.35 |
| 79 | 13779 | 104.36 | 13644 | 8.09 | 13779 | 132.84 | 13712 | 23.01 |
| 80 | 13614 | 184.91 | 13717 | 13.65 | 13606 | 164.04 | 13606 | 150.7 |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 81 | 13155 | 74.37 | 13109 | 11.71 | 13155 | 55.86 | 13195 | 65.91 |
| 82 | 13986 | 35.53 | 13979 | 15.17 | 13986 | 24.29 | 13986 | 27.46 |
| 83 | 13263 | 122.13 | 13243 | 11.88 | 13263 | 114.18 | 13240 | 67.73 |
| 84 | 13754 | 33.23 | 13710 | 7.77 | 13754 | 32.88 | 13715 | 138.62 |
| 85 | 13301 | 48.17 | 13194 | 6.26 | 13301 | 43.81 | 13273 | 138.34 |
| 86 | 14483 | 110.51 | 14443 | 8.14 | 14483 | 103.65 | 14544 | 152.88 |
| 87 | 13122 | 91.57 | 13150 | 11.74 | 13116 | 53.45 | 13115 | 59.96 |
| 88 | 13658 | 47.88 | 13559 | 6.87 | 13658 | 44.86 | 13584 | 127.06 |
| 89 | 13460 | 26.1 | 13497 | 11.4 | 13460 | 25.84 | 13595 | 112.5 |
| 90 | 14156 | 52.33 | 14117 | 12.97 | 14232 | 25.71 | 14214 | 23.33 |
| 91 | 14126 | 140.15 | 14101 | 17.48 | 14126 | 144.31 | 14093 | 170.07 |
| 92 | 13747 | 70.04 | 13648 | 15.04 | 13705 | 66.87 | 13686 | 80.52 |
| 93 | 13823 | 28.77 | 13764 | 13.44 | 13823 | 27.64 | 13768 | 142.08 |
| 94 | 13927 | 81.01 | 13889 | 13.85 | 13927 | 79.25 | 13916 | 66.63 |
| 95 | 13927 | 119.82 | 13888 | 13.4 | 13927 | 113.75 | 14017 | 180.84 |
| 96 | 13540 | 81.05 | 13575 | 11.3 | 13540 | 104.72 | 13540 | 94.26 |
| 97 | 13247 | 29.07 | 13196 | 12.12 | 13180 | 128.84 | 13182 | 93.69 |
| 98 | 13663 | 91.97 | 13618 | 11 | 13606 | 101.99 | 13606 | 91.88 |
| 99 | 15056 | 226.23 | 14919 | 18.12 | 15014 | 179.76 | 14960 | 282.83 |
| 100 | 13236 | 161.45 | 13242 | 10.13 | 13236 | 148.62 | 13330 | 134.05 |