Multi-objective Trip Planning with Solution Ranking Based on User Preference and Restaurant Selection

SUPOJ CHOACHAICHAROENKUL¹, DAVID COIT²,3 and NARUEMON WATTANAPONGSAKORN¹
¹King Mongkut’s University of Technology Thonburi, Bangkok, 10140 Thailand
²Rutgers University, Piscataway, NJ, 08854, USA
³Tsinghua University, Beijing, China

Corresponding author: Naruemon Wattanapongsakorn (e-mail: naruemon.wat@kmutt.ac.th)

This work is supported by Petchra Pra Jom Klao Doctoral Scholarship, grant number 15/2560, King Mongkut’s University of Technology Thonburi.

ABSTRACT The tourist trip design problem (TTDP) helps the trip planners, such as tourists, tour companies, and government agencies, automate their trip planning. TTDP solver chooses and sequences an optimal subset of point of interest (POIs), which adhere to the POIs attributes and tourist preferences, and then generates a travel itinerary that maximizes their pleasure. However, the traditional TTDP does not include the lunch period at a local restaurant, which causes the rest of the itinerary in the afternoon to shift, nor compulsory POIs that the trip planners must be included in the itinerary. Moreover, as tourism contributes to high greenhouse gas emissions, especially from its transportation, minimizing the itinerary’s total distance is also considered. Unfortunately, this objective conflicts with the profit scores; no single itinerary can optimize both objectives simultaneously. Hence, the multi-objective technique and the results of non-dominated itineraries can be organized as a Pareto front. The trip planners can choose one suitable itinerary from the Pareto front based on their preferences. To address these real-world issues, we formulate a new variant of the well-known orienteering problem with time windows (OPTW) called the multi-objective orienteering problem with Time Windows, Restaurant Selection, and Compulsory POIs (MOPTW-RSCP). The proposed problem is provided with a mathematical formulation and two exact algorithms for solving them, i.e., greedy and branch-and-cut Pareto-based techniques. The algorithms’ performance is tested against the Rattanakosin island (the old city of Bangkok) dataset. We conduct 24 test cases, and the computational results confirm the algorithms’ efficiency.

INDEX TERMS branch-and-cut algorithm, greedy algorithm, knee solution, low carbon tourism, multi-objective optimization, orienteering with time windows, point-of-interests, recommendation, solution ranking, tourist trip design problem

I. INTRODUCTION

Tourists visit points of interest (POIs) in a limited time, and it is virtually impossible to stop by all attractions in a destination. Therefore, they need an itinerary that comprises a proper subset of POIs based on their preferences and constraints. Thus, travel itineraries benefit tourists and the entire tourism industry, such as tour guides, companies, and government agencies.

The tourist trip design problem (TTDP) is proposed to describe the issue. It takes a POI list, a POI-to-POI distance table, and a tour period as the inputs. Each POI has a satisfying level called profit score and amount of visiting period. Therefore, the solution of a TTDP is an itinerary that routes a subset of POIs together with the highest total profit that obeys the tour period. Many TTDPs also consider the operational hours of each POI (time window) as a constraint to reflect the real world.

However, making a travel itinerary is a complex, tedious, and time-consuming process. Moreover, it is an NP-hard problem that has gathered momentum for the last two decades. Many researchers recently proposed efficient
techniques to solve the problem by inventing techniques to generate optimal travel itineraries automatically.

TTDP is a very complex problem, and the requirements may be varying. Several modified models of TTDP have been proposed in the past decade to promote real-world needs. We found variants of TTDP models that attempted to capture the real world from the literature review, but there were some limitations. Firstly, only a few of them are aware of the lunch period. The ignorance of this issue may cause the whole POIs to visit in the afternoon to be forced to shift. In addition, the user preference of restaurant selection is not considered. Moreover, Some POIs are usually put in the itinerary by the tourists’ preferences themselves or the recommendation from some sources. Unfortunately, most of the existing TTDP models do not support these compulsory POIs.

Lastly, but most importantly, the tourism industry contributes very high carbon emissions at roughly 8% of all sources worldwide. About one half (4%) came from transportation [1]; it is a global issue. In November 2021, the UK hosted the 26th UN climate change conference of the parties (COP26), and the world tourism organization (UNWTO) declared, in the forum, to cut their emissions in half by 2030 and down to zero no later than 2050 [2]. Hence, the tourism industry will intensely focus on lower carbon emission and aims to archive carbon neutral tourism. Then, the international transportation workers’ federation (ITF) responded by stating in November 2021 that transportation currently produces a quarter of CO2 emissions. Instead of cars, they insist on the global shift to public transit, walking, and cycling [3]. Hence, we support the issue by removing the transportation needed from the tour as much as possible. A walking tour fits the purpose because it requires only public transit and mostly only at the beginning and the end of the tour time. Moreover, the trip planners can adapt some days in the existing multi-day itinerary to be walking tours, and carbon emission can be significantly reduced.

A walking tour has its characteristic that the long total distance in a walking tour may cause the tourists to be tired. Therefore, a trade-off between the total profit and distance is needed. However, the suitable amount of distance is hard to determine, and it varies from person to person and from time to time. In other words, there is no single suitable total distance that suits a tourist in every situation. Hence, the total distance should be explicitly considered as important as the total profit. Therefore, the multi-objective optimization model is embraced to capture both aspects equally.

The solution from a multi-objective optimization model is not a single best itinerary but a set of non-dominated itineraries that let the trip planners choose the most appropriate one. However, when the preferences are not available, a solution ranking technique can support the decision.

This paper proposes a multi-objective orienteering problem with time windows, restaurant selection, and compulsory POIs (MO-OPTW-RSCP) based on user preferences to solve the issues above. The main contributions of this paper are as follows.

1. A real-world MO-OPTW-RSCP model is formulated and introduced for the first time.
2. Compulsory POIs based on user preference are provided.
3. Restaurant selection for lunch based on user preference.
4. A multi-objective branch-and-cut algorithm is proposed.
5. A multi-objective greedy-based algorithm is proposed.
6. A solution ranking technique to help trip planners choose outstanding itineraries from the Pareto front containing a set of Pareto-optimal solutions.

The rest of the paper is organized as follows. Background and related works are reviewed in Section II. Next, the mathematical model of MO-OPTW-RSCP along with the dataset is described in Section III. Then, the multi-objective branch-and-cut and greedy algorithms are proposed in Section IV. Then, Section V describes the benchmark instances, parameter settings, and evaluation metrics. The results obtained by both algorithms are compared using performance metrics, and the solutions are ranked. Finally, Section VI covers the conclusions and future work.

II. BACKGROUND AND RELATED WORKS

A. ORIENTEERING PROBLEM

The tourist trip design problem (TTDP) can be viewed as an application of the orienteering problem (OP) [4]. The OP was originated from a sport that competitors can choose to visit the places with various profit scores in the forest. The winner is the one who can collect the highest total profit from a subset of locations within a limited period. In 1984, Tsiligridis [5] formalized the problem by mathematically formulating the problem. The problem is an NP-hard problem. Several well-known variants of the problem have been modeled, including team OP (TOP), OP with time window (OPTW), team OPTW (TOPTW), time-dependent OP (TDOP), TDOP with time window (TDOPTW), and team TDOPTW (TD-TOPTW). The models have been comprehensively reviewed by Vansteenwegen and Gunawan in 2019 [6].

B. OP FOR TTDP

TTDP can be mapped to the OP by replacing visiting places and competitors with POIs and tourists. Furthermore, to make the itineraries more practical, time windows are often considered as each POI has its own open and close time. Hence, the most popular models of TTDP are based on OPTW for one-day itineraries and TOPTW or multi-day itineraries. In this paper, we will focus only on the OPTW for a single-day trip.

TTDP itself has several specified models which have been proposed to capture some unique real-world aspects. For example, Divsalar, et al. [7] presented OP with hotel selection...
(OPHS) to choose the hotel that makes the cost of traveling lowest [8-10].

Having lunch for tourists is also vital for many trip planners to specify a restaurant in the itinerary explicitly for reasons such as the preferences of the tourists, beliefs and religions, and limitations such as food allergy. The original paper of TTDP by Vansteenwagen and Oudheusden [4] has also considered having lunch in a restaurant. Then, Vansteenwagen et al. [11] formalized the concept by designing the lunch period by enabling virtual POIs without locations. Tenemaza et al. [12] assigned the lunch period at the current POI, assumed that every POI has a place to eat, and allowed the late lunch period if the current POI is finished, then optimized for the shortest time of late lunch. Finally, Expósito et al. [13] proposed the model TTDP with clustered POIs (TTDP-clu) that separate POIs into several categories, including a restaurant. The model supports restaurant selection and lunch period by specifying the constraints of the restaurant category.

Some POIs are preferable compared to the rest. For example, some POIs highlight that everyone should visit first-time tourists, or some POIs match the trip planners’ preferences. Consequently, some POIs become compulsory POIs, so they must be included in the itineraries. The concept was introduced into OP by Gendreau et al. [14]. Palomo-Martínez et al. [15] model the OP with mandatory visits and conflict (OPMVC). The model supports both compulsory and conflicted POIs. The conflicted POIs are those POIs that can be selected only one. For example, only one temple can be added to the itinerary. In the same year, Palomo-Martínez et al. [16] solved the OP problem by using VNS and GRASP techniques. Li and Hu [17] merged OPTW with compulsory POIs and modeled it as the OP with compulsory nodes and time windows (OPCPTW). Lin and Yu [18] brought the compulsory POIs to the TOPTW model and named it TOPTW with mandatory visits (TOPTW-MV).

**C. MULTI-OBJECTIVE OPTW**

Typically, the OP objective is to maximize total profit, but some models are concerned with more than one objective to be optimized. Then, they are classified as multi-objective optimization models with conflicting goals. Popular techniques to address with this problem are to merge multiple objectives into a single objective, hierarchical orders of the objectives, or using a Pareto-based method. The first two techniques result in a single best solution, whereas the last technique results in multiple non-dominated solutions. In this paper, we only consider the Pareto-based model.

The Pareto-based multi-objective optimization is based on Pareto dominance which states that if itinerary A has at least one objective superior to itinerary B with no worse objective, then we can say that itinerary A dominates itinerary B, and itinerary B is then eliminated. When we perform pairwise tournament comparisons, only the non-dominated itineraries are left. They are the solutions to the problem, which can be visualized as a Pareto front; the plot with the axes are objectives. Fig. 1 depicts the concept of Pareto front for a specific TTDP, which has two objectives: total profit (the higher, the better) and total distance (the lower, the better). The Pareto dominance’s concept of an itinerary is demonstrated in the figure as the dominance area formed by an itinerary and the nadir point at the top-left corner of the graph. All itineraries within the region are eliminated. Hence, only two solutions, non-dominated itineraries, are left and formed a Pareto front. It is noted that the figure is one of our test cases with 13 POIs and three compulsory POIs, which contains 7,607 itineraries.

The concept of the Pareto front was emerged by Srinivas and Deb [19]. Since then, it has become a research focus. Nowadays, several state-of-the-art algorithms have been proposed including VEGA [20], MOEA/D [21], and NSGA-II [22].

Objectives for TTDP may be varied. One of the most well-known objectives is multi-score for each POI, such as level of interest, relaxation, readiness, entrance fees, etc. Schild et al. [23] proposed a generic bi-objective POI scores model for OP and the result as a Pareto front. The contributed papers on this type of problem are Martí et al. [24], Purevsuren et al. [25], Rezki and Aghezzaf [26], Aghezzaf, and Rezki [27], Martín-Moreno and Vega-Rodríguez [28], Rezki and Aghezzaf [29]. The [T]OPTW variant of this type of problem was proposed by Chen et al. [30] and Hu et al. [31].

Minimizing the total distance of the trip may be implicitly accomplished by using local search. However, bringing the trade-off itineraries with varying satisfaction total profit (maximizing) and total distance (minimizing) benefits the trip planner to balance. Unfortunately, there are only a few papers that address the problem. They are Karimi and Bashiri [32], Hapsari et al. [33], and Falco et al. [34] [35] applied TOPTW with five objectives: maximize profit score, number of preferred POIs of the tourists, number of recommended POIs, time of the tour, and minimize the total distance. On the other hand, Mirzaei et al. [36] attempted to balance the satisfaction level for each day. Hence the objectives are to maximize the score, minimize the difference between the highest and lowest scores for a TOP model.

Table I presents the summary of the related works and the proposed model. We focus on four entities: lunch period, restaurant selection, compulsory POIs, and bi-objective of total profit and distances. The table reveals a research gap that none of the existing multi-objective models address the lunch period, restaurant selection, compulsory POIs, or solution ranking. Hence, the proposed model is proposed to fill in these gaps.

**D. ITINERARY SELECTION FROM PARETO FRONT**

The trip planners may select only one outstanding itinerary to implement from the given Pareto front, which depends on their preference. The two apparent preferences available in trip planning are the itinerary with maximum profit and minimum distance. The first one is the maximum total profit itinerary.
which is often selected when the tourists want to explore the highlighted POIs, as many as possible, in the target destination, especially for the first time visiting. Moreover, the travel agents also organize familiarization trips for their partners to see as many as possible of the highlighted POIs. Then, the partners can design an itinerary that matches their customers’ needs. The second one is the minimum total distance which is often selected when the tourists consist of elders or young children in tours.

Figure 1. The concept of the Pareto front.

TABLE I
SUMMARY OF RELATED WORKS AND THE PROPOSED MODEL

| Base model | references | Lunch | Restaurant | Compulsory | Objective 1 | Objective 2 |
|------------|------------|-------|------------|------------|-------------|-------------|
| OP         | 1998 [14], 2017 [15], 2017 [16] | ✓     | ✓          | ✓          | P           | P1, P2      |
|            | 2009 [23], 2015[24], 2015 [25], 2017 [26], 2018 [27], 2018 [28], 2020 [29] |       |            |            |             |             |
| OPTW       | 2007 [4]   | ✓     | ✓          | ✓          | P           | P1, P2      |
|            | 2011 [17]  |       |            | ✓          | P           |             |
|            | 2016 [30]  |       |            |            |             |             |
| The proposed model | ✓ ✓ ✓ | ✓     | ✓          | ✓          | P           | (D)         |
| TOPTW      | 2011 [11]  | ✓     | ✓          | ✓          | P           |             |
|            | 2019 [13]  | ✓     | ✓          | ✓          | *           | *           |
|            | 2015 [34], 2016 [35] |       |            |            |             |             |
|            | 2018 [31]  |       |            |            |             |             |
|            | 2018 [32], 2019 [33] |       |            |            | P1          | P2          |

The works with blanked objective2 are single-objective and an objective in parenthesis is to minimize and otherwise is to maximize P, P1, P2, and D denote total profit, profit1, profit2 and distance, respectively.

* The two papers have five objectives.
Selecting an outstanding itinerary from a Pareto front when no preference is available is not a trivial task as virtually all itineraries on the Pareto front dominate the inferior itineraries, i.e., they are non-dominated itineraries. A pruning technique by Petchrompo et al. [37] performed a clustering technique and discovered an outstanding solution for each cluster. However, many researchers agree on an outstanding, well-balanced trade-offs and diversity solution called knee solution. The concept of knee solutions can be depicted for a convex shape of a Pareto front of both objectives: total profit and distance Fig. 2. The figure illustrates three knee itineraries on the Pareto front.

![FIGURE 2. Knee itinerary concept.](image)

The three most popular techniques for finding knee solutions are the maximum distance from the hyperplane focus, utility-based focus, and angle-based focus. The maximum distance from the hyperplane focus was firstly described by Das [38]. First, Das defined a hyperplane that connects the extreme solutions. Then, the solution with the most extended length from itself perpendicular to the hyperplane is the knee solution. Several knee searching algorithms are based on this technique, e.g., [39-41].

The utility-based focus was introduced by Branke et al. [42] based on the trade-off of objective functions. The knee solution has a small gain of an object that causes the highest loss of the other objective, e.g., [43-45].

The angle-based focus was also seeded by Branke et al. [46]. The typical Pareto dominance can be geographically viewed as the area of domination to be the shape of rectangular, so it is the right angle at the solution. However, if the angle is more expansive, the scope of domination is extended. Therefore, some non-dominated solutions are eliminated, and the rest are considered as knee solutions, e.g. [47-49].

The knee solution acts as decision support, not a mandatory itinerary to select in the real world. Chooachaicharoenkul and Wattanapongsakorn [50] provide the angle-based focus solution ranking that will rank the solutions on a Pareto front to empower the trip planners.

III. PROBLEM DESCRIPTION AND FORMULATION

This section proposes the Multi-objective Orienteering Problem with Time Windows, Restaurant Selection, and Compulsory Pois (MOPTW-RSCP), which extends the Orienteering Problem with Time Windows (OPTW). We focus OPTW on the Tourist Trip Design Problem application (TTDP) for a one-day trip with time windows. MOPTW-RSCP extends the OPTW in several aspects, which are described below.

Firstly, the model has two objectives: maximizing the total profit and minimizing the total distance of the selected POIs in the itineraries. So, the model becomes a multi-objective model. However, as the typical characteristic of the multi-objective model, the objectives conflict. In this case, the total distance conflicts with the total profit; therefore, no single itinerary typically can satisfy both objectives simultaneously. The solution to this problem will be a set of trade-offs between non-dominated itineraries. If a tourist focuses on the high total profit, the distance tends to be longer and vice versa. The solution set can be visualized as a Pareto front, which is each axis representing an objective. Typically, the trip planners need only an itinerary from the solution set to fit their requirements.

Secondly, the model includes an explicit lunchtime and a small set of lunchtime restaurants. The list of restaurants may be from various options, e.g., tourists’ preferences or tour companies’ contracts. Thirdly, compulsory POIs are supported in the model.

Finally, most TTDP models allow to wait at the POI before its open time for relaxing the restrictions, and more variety of itineraries can be obtained. Our model also supports this concept by adding the maximum waiting time constraint independently for each POI, which can serve two purposes. Firstly, it will help the tourists to not wait too long for each POI. Secondly, it will help trip planners to not end the itinerary too early by setting the open time of the end depot to the tour end time. Therefore, the itinerary will span near the tour end time. Then, the tourists can utilize their allocated tour period efficiently.

A. MATHEMATICAL MODEL

Assuming that \( n \) attractions are in the destination, the attractions are separated into three types: point of interest (POI), restaurant, and start and end depots. Then, the attractions are encoded into an array that can be depicted in Fig. 4. \( S_p \) and \( E_p \), \( S_r \) and \( E_r \), and \( S_d \) and \( E_d \) keep track of the start and stop indexes of POIs, restaurants, and depots.
respectively. Then, you can browse the meaning of symbols quickly using TABLE II for the list of parameters and TABLE III for the list of decision variables.

### TABLE II

| Symbol | Meaning |
|--------|---------|
| $S_p, E_p$ | Start and end index of POIs |
| $S_r, E_r$ | Start and end index of restaurants |
| $S_h, E_h$ | Start and end index of depots |
| $T_{min}, T_{max}$ | Start and end time of the tour |
| $b, e$ | Attraction index; $b, e = S_p, ..., E_h$ |
| $N_p$ | Number of POIs. |
| $N_m$ | The number of compulsory POIs. |
| $N_a$ | The number of attractions. |
| $W_{S_p}, ..., W_{E_h}$ | The amount of waiting time allowed before the open time of the attraction |
| $P_{S_p}, ..., P_{E_p}$ | The profit score of each POI |
| $O_{S_p}, ..., O_{E_p}$ | Start operating time of each POI |
| $C_{S_p}, ..., C_{E_p}$ | Close operating time of each POI |
| $J_{S_p}, ..., J_{E_h}$ | Time used in each attraction |
| $M_{1}, ..., M_{N_m}$ | Compulsory POIs |
| $V_{b,e}$ | Travel cost from attraction $b$ to attraction $e$ (minutes) |
| $D_{b,e}$ | Distance cost from attraction $b$ to attraction $e$ (meters) |

### TABLE III

| Symbol | Meaning |
|--------|---------|
| $x_{b,e}$ | One if attraction from attraction $b$ to attraction $e$ is included in the itinerary or zero otherwise |
| $a_{S_p}, ..., a_{E_h}$ | Arrival time at each attraction |

MOPTW-RSCP has two objectives, one for maximizing total profit and the other for minimizing the total distance, that can be represented as follows:

Maximize \[ \sum_{i=S_p}^{E_p} x_{i,E_p} \sum_{j=S_p}^{E_p} P_j x_{i,j} \] (1)

Minimize \[ \sum_{i=S_p}^{E_p} x_{i,E_p} \sum_{j=S_p}^{E_p} D_j x_{i,j} \] (2)

subject to

\[ \sum_{i=S_p}^{E_p} x_{i,E_p} = \sum_{i=S_p}^{E_p} x_{i,D_0} = 1 \] (3)

\[ \sum_{i=S_p}^{E_p} x_{i,k} = \sum_{i=S_p}^{E_p} x_{k,j} \leq 1; \forall k = S_p, ..., E_h \] (4)

\[ O_i - W_i \leq a_i \leq C_i - J_i; \forall i = S_p, ..., E_h \] (5)

\[ a_{S_p} = T_{min} \] (6)

\[ a_i + V_{i,j} - a_j \leq \infty \times (1 - x_{i,j}) \; \forall i, j = S_p, ..., E_h \] (7)

\[ T_{min} + \sum_{i=S_p}^{E_p} \sum_{j=S_p}^{E_p} V_{i,j} x_{i,j} \leq T_{max} \] (8)

\[ \sum_{i=S_p}^{E_p} x_{i,k} = \sum_{j=S_p}^{E_p} x_{k,j} = 1; \forall k = S_p, ..., E_h \] (9)

\[ \sum_{i=S_p}^{E_p} x_{i,k} = \sum_{j=S_p}^{E_p} x_{k,j} = 0; \forall k = S_p, ..., E_h \] (10)

Constraint (3) ensures that the itinerary starts at the start depot and ends at the end depot. Each place is visited a maximum of once. Constraint (4) forces the chance to include a POI into the itinerary at most once. Constraint (5) states that the arrival time to each POI must be within its time window or within $z_3$ minutes before the open time. Constraint (6) sets the tour start time at $T_{min}$. Constraint (7) ensures that the path between two attractions is valid if the path is included in the itinerary. Constraint (8) provides that the itinerary agrees with the time budget. Constraint (9) and (10) ensure that the itinerary must include exactly one restaurant. Constraint (11) provides that all compulsory POIs will be included in the itinerary. Finally, constraint (12) ensures that no POI is connected to itself.

### B. CASE STUDY AREA AND DATASET

The research is intended to match the real-world problem as much as possible. So, the actual destination of Rattanakosin Island, the old city of Bangkok, Thailand, is selected. The area is 4.1 square kilometers in size, founded in 1782 as Siam’s center of power. Now, the site is one of the most magnificent places in the world.

We collected 50 tourist attraction POIs along with their geographic locations and operational hours from Google Map. Then, we asked a tour company to fill up the times used in minutes and profit scores in the range of 1 to 100 based on popularity for each POI. Finally, we also asked for three recommended POIs and restaurants, and we obtained: the Grand Palace, Wat Pho (Pho temple), and flower market, along with five restaurants that we will use as the compulsory POIs and the lunchtime restaurants, respectively. As a result, the map of the island is shown in Fig. 3. From the figure, a circle denotes a POI, where the bigger size means more profit, the unfilled circles represent the compulsory POIs from top to bottom: Grand palace (C1), Wat Pho (C2), and flower market (C3), respectively. A square denotes a restaurant, and the triangle indicates the Sanamchay subway station, which is the start and the end of the tour.

The area is ideally suited for a walking tour. So, we set the start and end depots to Sanamchay Station, the just-operated subway station. The tourists can board the mass transit to visit the area from other parts of Bangkok and its suburb. We calculated the distance matrix of the walking routes using Openrouteservice and converted it to minutes-used at the leisure pace of 2.2 km/hour. Hence, the POIs can be visualized as Fig. 4. Triangles, circles, and squares denote the depot, POIs, and restaurants, respectively. The three unfilled circles (rings) indicated the compulsory POIs, from top to bottom: Grand Palace, Wat Pho, and the flower.
market, respectively. It is noted that the size of the circles for each POI denotes its profit score; the bigger, the higher the profit score.

![Figure 3](image-url) The map of Rattanakosin island.

![Figure 4](image-url) The structure of Attractions.

To analyze the performance of the proposed algorithms, we separated the dataset into smaller sets of 13, 15, 20, 30, 40, and 50 POIs. We need every dataset to contain all compulsory POIs. So we shuffle all POIs randomly until the first 13 POIs contain all three compulsory POIs. Then, the bigger datasets are also filled up with the three compulsory POIs and the rest of shuffled POIs. The technique can guarantee that every sub-dataset contains all three compulsory POIs.

The example of an itinerary is depicted as the walking route in Fig. 5 and the detailed itinerary in Table IV. The figure and the table demonstrate solution A from section IIIC. The route from Sanamchai station starts at 8:00, then visits several places, including compulsory POIs and a restaurant for lunch, and returns to Sanamchai Station before 17:00. The figure reveals the order of sight-seeing, from zero at Sanamchai Station and then one at drum tower and so on, then back to Sanamchai as the final POI. From the table, we can conclude that the routes’ total profit and distances are 370 and 6,665 meters, respectively.

![Figure 5](image-url) The route of solution A.

### TABLE IV

| ID. | Walk distance (mins) | Time     | POI               | visit (mins) | profit (score) |
|-----|----------------------|----------|-------------------|--------------|----------------|
| 0   | -                    | 8:00     | Sanamchai station | -            | -              |
| 1   | 6                    | 8:06     | drum tower        | 15           | 30             |
| 2   | 9                    | 8:30     | Pho temple (C2)   | 90           | 90             |
| 3   | 28                   | 10:28    | Grand palace (C1)| 90           | 100            |
| 4   | 22                   | 12:20    | Mattayaram temple| 30           | 30             |
| 5   | 10                   | 13:00    | Lunch: Ku wa aroi| 60           | -              |
| 6   | 17                   | 14:17    | Phut center       | 25           | 20             |
| 7   | 23                   | 15:05    | Ong Ang street    | 30           | 30             |
| 8   | 27                   | 16:02    | flower market (C3)| 20           | 70             |
| 0   | 11                   | 16:33    | Sanamchai station | -            | -              |
|     |                      |          |                   | 6665         | 370            |
traverse down the tree’s deeper levels if any of the following conditions are met.

- The lunch period is over, but no restaurant is included in the itinerary.
- A POI with the closing time is overdue.
- There is not enough time to append the compulsory POIs to the itinerary.
- A restaurant if the lunch period is fulfilled.

When a valid itinerary is found, the algorithm will perform the Pareto-dominance testing for the newly found itinerary against the current solution set of itineraries to guarantee that the solution set always keeps the non-dominated itineraries. Algorithm 1 implements the above concept by eliminating the itineraries in the current solution set; if the new itinerary dominates them, the new itinerary will be appended to the solution set. However, the algorithm will not insert the itinerary into the solution set if any itinerary in the solution set can dominate it. The eliminate formula is at line 4 which each relational operator results in one for true and zero for false. Thus, the new itinerary dominates in the solution set if \( r \) is positive and the reverse is negative, but both are non-dominated solutions if \( r \) is zero.

### TABLE V

| field     | datatype | description                  |
|-----------|----------|------------------------------|
| \( p \)   | array of integers | the order of POIs          |
| \( f \)   | array of integers | the candidates POIs to be inserted |
| \( \hat{p} \) | integer | accumulation of the profits |
| \( d \)   | integer | accumulation of the distances |
| \( l \)   | boolean | there is a restaurant in the path or not |
| \( t \)   | integer | the current time of the last appended POI to the path. |
| \( \hat{\hat{c}} \) | integer | the total of time used for all compulsory POIs. |

#### Algorithm 1 ParetoDominance

**Input:** nondominated solution set \( ss \) and a valid itinerary \( it \)

**Output:** nondominated solution set

1. \( isND \leftarrow true \)
2. for \( i \) in \( size(rs) \) do down to 1
3. \( isND \leftarrow true \) and break if \( sort(it, p) = sort(ss, p) \)
4. \( r \leftarrow (it.\hat{p} > ss.\hat{p}) + (it.\hat{d} < ss.\hat{d}) + (ss.\hat{p} > it.\hat{p}) \)
5. \( ss \leftarrow ss \cup it \) if \( r > 0 \)
6. \( isND \leftarrow true \) and break if \( r < 0 \)
7. end
8. \( ss \leftarrow ss \cup it \) if \( isND = true \)
9. return \( ss \);

#### Algorithm 2 BranchAndCutMain

**Input:** All Parameter from table 11

**Output:** nondominated solution set \( ss \)

1. \( ss \leftarrow \{ \} \)
2. \( it \leftarrow \text{new itinerary structure} \)
3. \( it.p \leftarrow \{ sa \} \)
4. \( it.f \leftarrow \{ 1, \ldots, N_3 \} \) \(-\{ sa \} \)
5. \( it.\hat{p} \leftarrow it.\hat{d} \leftarrow 0 \)
6. \( it.\hat{\hat{c}} \leftarrow \sum_{i=1}^{m} t_i \)
7. \( it.\hat{t} \leftarrow T_{min} \)
8. \( it.l \leftarrow false \)
9. \( ss \leftarrow \text{BranchAndCutRecursive}(it, rs) \)
10. return \( ss \);

#### Algorithm 3 BranchAndCutRecursive

**Input:** an uncompleted itinerary \( it \), solution set \( ss \)

**Output:** nondominated solution set \( ss \)

1. if \( it.\hat{P} = E_h \)
2. ParetoDominance(\( ss, it \) if \( it.l = true \) and \( it.\hat{c} = 0 \)
3. return \( ss \)
4. end if
5. return \( ss \) if \( (it.l \geq T_{end}) \) and \( it.l = false \) or \( T_{end} = it.\hat{t} \)
6. for each \( f \) in \( it.f \)
7. continue if \( not(0_f - W_f \leq it.\hat{t} + V_{it.\hat{P}} \leq C_f) \)
8. \( ff \leftarrow it.f - f \)
9. \( ff \leftarrow ff - \{ S_0, \ldots, E_k \} \) if \( f \in \{ S_0, \ldots, E_k \} \)
10. \( t2 \leftarrow \text{new itinerary structure} \)
11. \( it2.d \leftarrow \text{it}.d + D_{it.\hat{P}} \)
12. \( it2.p \leftarrow \text{append}(it, p, f) \)
13. \( it2.f \leftarrow ff \)
14. \( it2.l \leftarrow it.l \) \( \forall f \in \{ S_0, \ldots, E_k \} \)
15. \( it2.\hat{p} \leftarrow it.\hat{p} + P_f \)
16. \( ss \leftarrow \text{BranchAndCutRecursive}(it2, ss) \)
17. end for
18. return \( ss \);

### B. GREEDY ALGORITHM

The greedy algorithm builds up the itineraries by filling them with higher profit POIs into all starter paths segments made from each restaurant’s permutation and compulsory POIs. The recursive technique is employed to fill up a POI per level of calling, and the algorithm will stop traversing down if a valid itinerary is found. Thus, the algorithm is fast and can
obtain high-quality itineraries; it cannot guarantee optimal itineraries.

The main algorithm is shown in Algorithm 4. It builds the permuted itineraries that contain only essential POIs: a restaurant and compulsory POIs. The process is done in lines 3-5. Consequently, for each sequence of POIs \( p \), the corresponding unassigned POIs (\( f \)) are assigned and sorted by high-to-low profits for being injected into \( p \). Line 7, Evaluate function converts \( p \) into an itinerary and then feed to the core recursive algorithm.

Algorithm 5 depicts the core recursive algorithm that injects each POI from \( f \) into the given itinerary. Line 1, if the current itinerary is completed, no deeper level is traversed, the ParetoDominance function is called to attempt to insert it into the solution set. After that, the position of the restaurant (\( r \)) is determined in line 3, and attempts to find the best orders to be inserted and traversed down for the before noon period (Lines 4 and 5) and afternoon period (Lines 6 and 7). Line 8 and 9, The algorithm also ignores the current best unassigned POI for giving the chance to produce the itineraries without that POI.

Algorithm 6 computes the variables from the structure of an itinerary (Table V) from a given POI list. The algorithm also determines the POI list that is valid, semi-valid, or invalid. A valid itinerary is a usable itinerary, whereas the invalid itinerary violates the constraints: reach a POI after its close time (line 6), more than one restaurant (line 10). Moreover, the number of compulsory POIs is not fulfilled (line 15), but no semi-valid POI is found, also considered an invalid itinerary. The semi-valid is when the constraint of the open time of a POI is not satisfied; the POI is reached too early. For this case, it is possible to insert another POI before this one (line 7). The main loop in lines 5-14 considers each POI in the POI list and gathers \( nc \) the number of compulsory POIs, \( nr \) the number of restaurants, and the variables in the itinerary structure.

Algorithm 7 converts a sequence of POIs into an itinerary. Loop in lines 5-14 will try to insert the current highest profit (\( f_1 \)) into each segment and the valid itinerary with the shortest total distance will be returned to the caller.

The local search is presented in Algorithm 8. It attempts to replace a POI in each itinerary of the solution set with a POI in the unassigned POIs that minimizes the total distance. Lines 1-3 are for looping all POIs in each itinerary in the solution set. The POIs must not be the compulsory POIs nor the restaurant; the checking is performed in line 4. Then, the algorithm will try to replace the POIs with the POIs with equal profits from the unassigned POIs list; A POI that makes the lowest total distance will be replaced with the original one. Finally, the algorithm will return the local search solution set in line 15.
Algorithm 8 Local Search

| Input: solution set ss | Output: solution set |
|------------------------|---------------------|
| for it in ss           |                     |
| best ← copy(it)        |                     |
| for i in 2,...,[it.p] − 1 |                 |
| continue if(it.p)<∈{Sr,...,Ei,Me,...,Ml} |      |
| for j in Sr,...,Ei     |                     |
| continue if j<∈{it.p} or Pj≠pit.p |          |
| p ← copy(it.p)         |                     |
| ti2 ← Evaluate(p)     |                     |
| best ← it2 if best.δ < it2.δ |             |
| end for                |                     |
| end for                |                     |
| replace it with best in ss |                   |
| return ss              |                     |

C. SOLUTION RANKING

A Pareto front comprises several itineraries that enable the trip planners to have more choices to choose. They can choose the most suitable itinerary based on their preferences. For example, if they want to enjoy the trip as much as possible, they may select the itinerary with the highest total profit or choose the lowest total distance if they do not want to spend much time walking. Unfortunately, choosing the most suitable itinerary might not be a trivial task, primarily when they do not provide a preference.

To solve this problem, we embrace the concept of knee solutions, the widely accepted technique when no preference is provided. Moreover, knee-based solution ranking is employed to highlight outstanding itineraries and offer trip planners more choices. For this reason, we used RADA [50], an angle-dominance knee-based solution ranking algorithm, to rank the itineraries from the Pareto fronts that both previous sub-sections have generated. However, we have post-processed the result from RADA not to index the extreme solutions as they are always good choices for the trip planners to be considered.

V. EXPERIMENTS AND RESULTS

A. BENCHMARK INSTANCES AND PARAMETER SETTING

In the experiments, we produce six subsets from Rattanakosin island, as mentioned in Section III.B, for 13, 15, 20, 30, 40, and 50 POIs and four sets of the compulsory POIs:

- None (C0)
- The Grand Palace (C1)
- The Grand Palace (C1) and Pho Temple (C2)
- The Grand Palace (C1), Pho Temple (C2), and Flower market (C3)

Therefore, 24 benchmark instances for each algorithm are prepared. The objectives for each itinerary are maximizing the total profit and minimizing the total distance.

The parameters of the environments are described in Table VI. It is noted that the open and close time at the end depot is set to tour end time. So, each itinerary is forced to be completed no earlier than W_{Eh} minutes from the tour end time.

| Variable | Meaning | Value |
|----------|---------|-------|
| T_{min},T_{max} | tour time | 8:00-17:00 |
| L_{min},L_{max} | lunch period | 11:30-13:00 |
| W_{S1,...,S5} | wait time allowed before a POI | 15 mins |
| W_{R1,...,R5} | wait time allowed before a restaurant open | 15 mins |
| W_{Eh} | Amount of time allowed to finish the tour before end tour time | 30 mins |

B. EVALUATION METRICS

This research chooses the top three indicators to evaluate the greedy algorithm against the benchmarked branch-and-cut algorithm from the standard indicators for multi-objective optimization [51]. These indicators are probably the most accepted indicators for Pareto-based multi-objective optimization. First, we pre-processed the results by negating the total profit to minimize both objectives. Then the two objectives were normalized to the values between zero and one. The G and E denote the set of greedy and branch-and-cut solution sets, respectively.

1) Hypervolume indicator (HV)

Hypervolume [52], in this research work, measures the area of the dominated portion of the objective space from the nadir point; it is the point (0, 0) for this case. It is also called S-metric. The higher values, the better the Pareto front. It is the most famous indicator, with 82% used in related multi-objective optimization papers [51]. The definition of HV is shown as Eq. (13). S can be G or E and p_i and d_i denotes the total profit and total distance of the solution i, respectively.

\[ HV(S) = p_i \cdot d_i \cdot \sum_{\{e2...\mid S - \{j}\mid = 1}\} (d_j - d_i) \cdot p_j \]  \hspace{1cm} (13)

2) Generational Distance (GD.)

Generational distance is defined as the average of the distance of each itinerary in the greedy solution set to its closest itinerary in the branch-and-cut solution set. Zero is the ideal solution that implies that two Pareto fronts are precisely the same. The definition of GD is shown as Eq. (14).

\[ GD(G,E) = \sum_{g \in G} \text{MIN}(\|g - e\|) / |G| \quad \forall e \in E \]  \hspace{1cm} (14)

3) IGD

The inverted generational distance (IGD) is the inverse version of GD, defined as the average of the total distance of each itinerary in the branch-and-cut solution set to its closest itinerary in the Greedy solution set. The concept of the values is the same as GD; Zero is the ideal value. The definition of IGD is shown as Eq. (15).

\[ GD(G,E) = \sum_{e \in E} \text{MIN}(\|e - g\|) / |E| \quad \forall g \in G \]  \hspace{1cm} (15)
C. NUMERICAL TESTS

Greedy and branch-and-cut, the two proposed algorithms were implemented using Julia 1.6 programming language on a desktop computer running on a Windows 11 with AMD Ryzen7 5700G 3.8MHz CPU and 16GB RAM. The time limit is set to 48 hours for each test case. If a test case cannot finish within the time limit, the time used record will be designated as “time out.”

The Pareto fronts of each test case are shown in Fig. 6 for 13, 15, and 20 POIs, and Fig. 7 for 30, 40, and 50 POIs. In each sub-figure of Fig. 6 shows the Pareto fronts of the greedy and branch-and-cut algorithms. While Fig. 7 shows only the Pareto front for the greedy algorithm because the time complexity of the branch-and-cut algorithm was too high to compute. For both figures, the x-axis represents total profit while the y-axis represents the total distance; the cross and circle signs represent the itineraries of the greedy and branch-and-cut, respectively. The optimal point is at the bottom-right corner due to the maximization of the total profit and the minimization of the total distance.

The number of itineraries of each sub-figure decreases if the number of compulsory POIs rises. These POIs are the constraints to reduce the possibility of the itineraries. From these observations, the greedy algorithms tend to produce the number of itineraries for each experiment less than the branch-and-cut algorithm except for the compulsory POIs of [3, 9, 10]; both algorithms produce identical number itineraries. The sub-figures reveal that the greedy algorithm can obtain high-quality itineraries very close to the branch-and-cut method’s Pareto front.

![Figure 6. Pareto fronts of 13, 15, and 20 POIs test cases.](image-url)
The performance of the greedy algorithm can be evaluated by using performance metrics. Fig. 8 depicts the hypervolume values of both algorithms that separate the subplots for each number of POIs. The x-label and y-label of each subplot represent the compulsory POIs and hypervolume values, respectively. The subplots show that the greedy algorithm had performed almost perfectly on 13 POIs, perfect on 15 POIs, and very well on 20 POIs. However, at 30 POIs with the compulsory POI of [3], the HV is lower than its no compulsory POI, which means that the front is not fully optimized. Fig. 9 separates the experiments by compulsory POIs. It also confirms the above statements that at 20 POIs, the greedy algorithm cannot find the fully optimal Pareto fronts compared with HVs of the branch-and-cut algorithm. Moreover, at 50 POIs with compulsory POIs of [3] and [3, 9], the greedy algorithm cannot find optimal Pareto fronts, because the HVs are lower than their 40 POIs’ HVs.

Table VII summarizes the GD and IGD metrics and time complexity of both algorithms. The GD ranges between 0 and 0.1494, whereas IGD ranges between 0 and 0.964. Zero means perfect optimal Pareto front at 15 POIs with compulsory POIs of [3, 9, 10]. The highest GD and IGD are quite low, which means they are close to the Pareto fronts of the branch-and-cut algorithm. Furthermore, the greedy algorithm is better than the branch-and-cut algorithm when comparing the time complexity in every experiment. The branch-and-cut algorithm can perform acceptably at 13 and 15 POIs, but at 20 POIs, it took 15.43 hours for [3, 9, 10] and more than 20 hours for None, [3], and [3, 9] whereas the greedy algorithm took no more than one minute. The branch-and-cut algorithm failed to achieve the results within 48 hours for 30 POIs or more, but the greedy algorithm could find the results within 1.11 minutes. It is worth noting that the time complexity is raised when the number of POIs is increased whereas the time complexity of the greedy algorithm is raised when the number of compulsory POIs is decreased.

D. SOLUTION RANKING
The itineraries ranking for each test case is shown in Fig. 10. The Pareto front of each test case is the input to RADA, and...
the ranking numbers are shown below each itinerary; the lower number, the better. If available, we choose the best Pareto front from the branch-and-cut algorithm, or the greedy one is selected for each test case. It is noted that the test cases with only two itineraries do not have a number because they belong to extreme solutions, and no itinerary is left to rank.

To illustrate the concept, we choose a test case of 50 POIs and a compulsory POI of C1 (Grand Palace) as an example. The solutions from the greedy algorithm comprise seven itineraries on the Pareto front. Table VII shows the total profit and distance of each itinerary. Fig. 12 depicts the Pareto front with ranking. The figure depicts two regions: left and right, which are denoted by rectangles. Each region has its knee itinerary. Rada ranked the 1st itinerary at the left group and the 2nd itinerary at the right group. Hence, the two solutions are good candidates to present to the trip planners in order. Moreover, the left region has the ranking of 1st, 3rd, and 5th while the right region has the ranking of 2nd and 4th which reveals how well the RADA algorithm span ranking as the algorithm has taken the diversity of solutions into account. The details and the routes of the two itineraries are depicted in Table IX-X and Fig. 13-14, respectively. The 1st rank itinerary focuses on lower total distance, while the 2nd rank itinerary focuses on a higher total profit score.

![Figure 8](image1.png)

**FIGURE 8.** HV values for each test case based on number of POIs.

![Figure 9](image2.png)

**FIGURE 9.** HV values for each case based on compulsory POIs.

![Table VII](image3.png)

**TABLE VII**

| Compulsory POIs | GD | IGD | Time used hh:mm:ss |
|-----------------|----|-----|---------------------|
| C1              |    |     |                     |
| C1 & C2         |    |     |                     |
| C1, C2 & C3     |    |     |                     |
| None            | 13 | 0.0411 | 0.0695 | 00:00:01 | 00:00:15 |
| 15              | 0.0264 | 0.0551 | 00:00:04 | 00:00:04 |
| 20              | 0.0715 | 0.0721 | 00:00:51 | 21:12:51 |
| 30              | -   | -    | 00:01:11 | time out |
| 40              | -   | -    | 00:00:37 | time out |
| 50              | -   | -    | 00:00:28 | time out |
| C1: 3           | 13 | 0.0263 | 0.0411 | 00:00:01 | 00:00:13 |
| 15              | 0.0059 | 0.0059 | 00:00:01 | 00:00:33 |
| 20              | 0.1494 | 0.0964 | 00:00:11 | 21:24:40 |
| 30              | -   | -    | 00:00:20 | time out |
| 40              | -   | -    | 00:00:11 | time out |
| 50              | -   | -    | 00:00:07 | time out |
| C1 & C2: 3, 9   | 13 | 0.0263 | 0.0411 | 00:00:00 | 00:00:12 |
| 15              | 0.0059 | 0.0059 | 00:00:00 | 00:03:36 |
| 20              | 0.0480 | 0.0521 | 00:00:01 | 23:03:26 |
| 30              | -   | -    | 00:00:03 | time out |
| 40              | -   | -    | 00:00:01 | time out |
| 50              | -   | -    | 00:00:01 | time out |
| C1, C2 & C3: 3, 9, 10 | 13 | 0.0526 | 0.0432 | 00:00:00 | 00:00:12 |
| 15              | 0.0000 | 0.0000 | 00:00:00 | 00:03:36 |
| 20              | 0.0530 | 0.0654 | 00:00:01 | 15:43:18 |
| 30              | -   | -    | 00:00:02 | time out |
| 40              | -   | -    | 00:00:01 | time out |
| 50              | -   | -    | 00:00:01 | time out |
FIGURE 10. Pareto fronts for 13, 15, and 20 POIs.
VI. CONCLUSIONS AND FUTURE WORK

We introduced the new real-world tourist trip design problem called the multi-objective orienteering problem with restaurant selection and compulsory POIs (MOPTW-RSCP). The Ratanakosin dataset, the old town of Bangkok, is used as the benchmark for this work, with 24 test cases varying the number of compulsory and total POIs. Then, we proposed two exact multi-objective algorithms for solving the problem.

The branch-and-cut algorithm, by theoretical basis, can search for all optimized solutions; however, the time complexity is exponentially high. Therefore, we enforced a time limit for each test case to be 48 hours. The branch-and-cut algorithm can easily search for the optimal Pareto fronts for 13 and 15 POIs, but for 20 POIs, it took 15-23 hours runtime. Hence, it failed to find the solutions when the number of POIs is more than 20, which was half of all test cases. Conversely, the greedy algorithm could efficiently find solutions within the time limit; the worst test case took only 1.11 minutes. The results are impressive with two test cases that the algorithm can obtain the optimal Pareto fronts, and the rests are almost optimal.

We used RADA to rank the itineraries as a guideline for choosing the outstanding itineraries using the knee solution concept to support trip planners. However, we exclude the extreme itineraries as the trip planners commonly select them.

We will focus on the CO₂ emissions issue in our future work and will extend the concept to a multi-day itinerary. However, it is less practical if every day in a tour comprises only public transportation and walking. Hence, we will focus on the low carbon tour with the carbon compensation that
makes a carbon neutral tour. The dataset will be enhanced from a city to a region; hence the model will be naturally a multi-day trip of TOPTW. We also have a plan to research a multi-object evolutionary algorithm (MOEA) to solve this problem.

### TABLE VIII
TIME USED FOR GREEDY AND BRANCH-AND-CUT ALGORITHMS AGAINST RATANAKOSIN DATASET

| Ranking | Total Profit (scores) | Total Distance (meters) |
|---------|-----------------------|-------------------------|
| 1       | 430                   | 5,428                   |
| 2       | 530                   | 6,138                   |
| 3       | 440                   | 5,707                   |
| 4       | 510                   | 5,891                   |
| 5       | 460                   | 5,861                   |
| extreme | 360                   | 5,390                   |
| extreme | 550                   | 6,719                   |

![Figure 12. Solution ranking for 50 Pois with C1 compulsory POI.](image)

### TABLE IX
THE DETAILS OF THE ITINERARY OF the 1st Rank of POIs=50 and C1 compulsory POI.

| ID. | Walk (mins) | distance (meters) | Time | POI                  | Visit (mins) | profit (score) |
|-----|-------------|-------------------|------|----------------------|--------------|----------------|
| 0   | -           | -                 | 8:00 | Sanamchai station   | -            | -              |
| 1   | 13          | 545               | 8:13 | Pho temple           | 90           | 90             |
| 2   | 28          | 1,222             | 10:11| Grand Palace(C1)    | 90           | 100            |
| 3   | 30          | 1,301             | 12:11| Ku Wa Aroi (R)      | 60           | 0              |
| 4   | 2           | 88                | 13:13| Pho Suea shrine     | 25           | 50             |
| 5   | 8           | 346               | 13:46| Devasathan           | 20           | 50             |
| 6   | 4           | 191               | 14:10| Giant Swing         | 15           | 50             |
| 7   | 0           | 15                | 14:25| Suthat temple       | 45           | 70             |
| 8   | 28          | 1,226             | 15:38| Phahurat market     | 30           | 50             |
| 9   | 17          | 719               | 16:25| Flower market       | 20           | 70             |
| 0   | 11          | 485               | 16:56| Sanamchai station   | -            | -              |

![Figure 13. The route of the 1st Rank of POIs = 50 and C1 compulsory POI.](image)

### TABLE X
THE DETAILS OF THE ITINERARY OF 2nd Rank of POIs=50 and C1 compulsory POI.

| ID. | Walk (mins) | distance (meters) | Time | POI                  | Visit (mins) | profit (score) |
|-----|-------------|-------------------|------|----------------------|--------------|----------------|
| 0   | -           | -                 | 8:00 | Sanamchai station   | -            | -              |
| 1   | 13          | 545               | 8:13 | Pho temple           | 90           | 90             |
| 2   | 28          | 1,222             | 10:11| Grand Palace(C1)    | 90           | 100            |
| 3   | 30          | 1,301             | 12:11| Ku Wa Aroi (R)      | 60           | 0              |
| 4   | 2           | 88                | 13:13| Pho Suea shrine     | 25           | 50             |
| 5   | 8           | 346               | 13:46| Devasathan           | 20           | 50             |
| 6   | 4           | 191               | 14:10| Giant Swing         | 15           | 50             |
| 7   | 0           | 15                | 14:25| Suthat temple       | 45           | 70             |
| 8   | 28          | 1,226             | 15:38| Phahurat market     | 30           | 50             |
| 9   | 17          | 719               | 16:25| Flower market       | 20           | 70             |
| 0   | 11          | 485               | 16:56| Sanamchai station   | -            | -              |

![Total time](image)
Figure 14. The route of the 2nd Rank of POIs = 50 and C1 compulsory POI.

REFERENCES

[1] M. Lenzen et al., "The carbon footprint of global tourism," Nature Climate Change, vol. 8, no. 6, pp. 522-528, 2018.
[2] UNWTO, "Tourism united behind the Glasgow Declaration on Climate Action at COP26," ed. Glasgow, United Kingdom: UNWTO, 2021.
[3] I. T. W. F. (ITF), "Making COP26 Count," ed. London, United Kingdom: ITF, 2021.
[4] P. Vansteenwegen and D. Oudheusden, "The Mobile Tourist Guide: An OR Opportunity," OR Insight, vol. 20, pp. 21-27, 2007.
[5] T. Tsiligiridis, "Heuristic Methods Applied to Orienteering," J. Oper. Res. Soc., vol. 35, pp. 797-809, 1984.
[6] P. Vansteenwegen and A. Gunawan, "Orienteering Problems: Models and Algorithms for Vehicle Routing Problems with Profits," Switzerland: Springer, 2019.
[7] A. Divsalar et al., "A variable neighborhood search method for the orienteering problem with hotel selection," in International Journal of Production Economics, pp. 150-160, 2013.
[8] S. Sohrabi et al., "A Greedy Randomized Adaptive Search Procedure for the Orienteering Problem with Hotel Selection," Eur. J. Oper. Res., vol. 283, no. 2, pp. 426-440, 2020.
[9] A. Toledo et al., "A Hyper-Heuristic for the Orienteering Problem With Hotel Selection," IEEE Access, vol. 8, pp. 1303-1313, 2020.
[10] W. M. Zheng et al., "Using a heuristic approach to design personalized urban tourism itineraries with hotel selection," Tourism Manage., vol. 76, pp. 1-14, 2020.
[11] P. Vansteenwegen et al., "The City Trip Planner: An expert system for tourists," Expert Syst. Appl., vol. 38, no. 6, pp. 6540-6546, 2011.
[12] M. Tenemaza et al., "Improving Itinerary Recommendations for Tourists Through Metaheuristic Algorithms: An Optimization Proposal," IEEE Access, vol. 8, pp. 79003-79023, 2020.
[13] A. Exposito et al., "A fuzzy GRASP for the tourist trip design with clustered POIs," Expert Syst. Appl., vol. 127, pp. 210-227, 2019.
[14] M. Gendreau et al., "A branch-and-cut algorithm for the undirected selective traveling salesman problem," Networks, vol. 32, pp. 263-273, 1998.
[15] P. J. Palomo-Martinez et al., "Formulations for the orienteering problem with additional constraints," Ann. Oper. Res., vol. 258, no. 2, pp. 503-545, 2017.
[16] P. J. Palomo-Martinez et al., "A hybrid variable neighborhood search for the Orienteering Problem with mandatory visits and exclusionary constraints," Comput. Oper. Res., vol. 78, pp. 408-419, 2017.
[17] Z. Li and X. Hu, "The orienteering problem with compulsory nodes and time window," in ICSSSM11, pp. 1-4, 2011.
[18] S.-W. Lin and V. F. Yu, "Solving the team orienteering problem with time windows and mandatory visits by multi-start simulated annealing," Comput. Ind. Eng., vol. 114, pp. 195–205, 2017.
[19] N. Srinivas and K. Deb, "Multiobjective optimization using nondominated sorting in genetic algorithms," Evol. Comput., vol. 2, no. 3, pp. 221–248, 1994.
[20] J. Schaffer, "Multiple Objective Optimization with Vector Evaluated Genetic Algorithms," in Int. Conf. Genet. Algorithms, pp. 93-100, 1985.
[21] Q. Zhang and H. Li, "MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition," IEEE Trans. Evol. Comput., vol. 11, no. 6, pp. 712-731, 2007.
[22] K. Deb et al., "A fast and elitist multiobjective genetic algorithm: NSGA-II," IEEE Trans. Evol. Comput., vol. 6, no. 2, pp. 182-197, 2002.
[23] M. Schilde et al., "Metaheuristics for the bi-objective orienteering problem," Swarm Intell., vol. 3, no. 3, pp. 179-201, 2009.
[24] R. Martí et al., "Multiobjective GRASP with Path Relinking," Eur. J. Oper. Res., vol. 240, no. 1, pp. 54-71, 2015.
[25] D. Purevsuren et al., "Evolutionary multiobjective optimization algorithms with path relinking for Bi-Orienteering problem," in ICESSS, pp. 132-135, 2015.
[26] H. Rezki and B. Aghezzaf, "The bi-objective orienteering problem with budget constraint: GRASP_ILS," LOGISTIQUA, pp. 25-30, 2017.
[27] B. Aghezzaf and H. Rezki, "λ-GRASP with bidirectional path relinking for the bi-objective orienteering problem," Int. J. Logist. Syst. Manag., vol. 29, no. 4, pp. 455-475, 2018.
[28] R. Martín-Moreno and M. A. Vega-Rodríguez, "Multi-Objective Artificial Bee Colony algorithm applied to the bi-objective orienteering problem," Knowledge-Based Syst., vol. 154, pp. 93-101, 2018.
[29] H. Rezki and B. Aghezzaf, "A hybrid GRASP for solving the bi-objective orienteering problem," Int. J. Oper, vol. 39, no. 4, pp. 494-515, 2020.
[30] Y. H. Chen et al., "Multi-objective orienteering problem with time windows: An ant colony optimization algorithm," in TAAI 2015, pp. 128-135, 2016.
[31] W. Hu et al., "A multi-objective evolutionary algorithm based on decomposition and constraint programming for the multi-objective team orienteering problem with time windows," Appl. Soft. Comput., vol. 73, pp. 383-393, 2018.
[32] A. Karimi and M. Bashiri, "The bi-team orienteering problem with time window," Int. J. Serv. Oper., vol. 30, no. 2, pp. 203-225, 2018.
[33] I. Hapsari et al., "Solving multi-objective team orienteering problem with time windows using adjustment iterated local search," Journal of Industrial Engineering International, vol. 15, no. 4, pp. 679-693, 2019.
[34] I. De Falco et al., "A multiobjective evolutionary algorithm for personalized tours in street networks," Lecture Notes in Computer Science, vol. 9028, pp. 115-127, 2015.
[35] I. De Falco et al., "Optimizing Personalized Touristic Itineraries by a Multiobjective Evolutionary Algorithm," Int J Inf Technol Decis Mak, vol. 15, no. 6, pp. 1269-1312, 2016.
[36] M. H. Mirzaei et al., "Bi-objective version of team orienteering problem (BTOP)," in ICCKE 2017, pp. 1-7, 2017.
[37] S. Petchrompo et al., "Pruning Pareto optimal solutions for multi-objective portfolio asset management," European Journal of Operational Research, vol. 297, no. 1, pp. 203-220, 2022.
[38] I. Das, "On characterizing the “knee” of the Pareto curve based on Normal-Boundary Intersection," Struct Multidiscipl Optim, vol. 18, pp. 107-115, 1999.
[39] S. Chouchaicharoenkul and N. Wattanaongpaksorn, "Finding Outstanding Solutions for Multi-objective Optimization Problems," in ACM International Conference Proceeding Series, pp. 18-22, 2020.
[40] O. Cuate and O. Schütze, "Pareto Explorer for Finding the Knee for Many Objective Optimization Problems," Mathematics, vol. 8, no. 10, p. 24, 2020.
[41] G. Yu et al., "A Multiobjective Evolutionary Algorithm for Finding Knee Regions Using Two Localized Dominance Relationships," IEEE Trans. Evol. Comput., vol. 25, no. 1, pp. 145-158, 2021.
[42] E. Branke et al., "Finding knees in multi-objective optimization," in Parallel Problem Solving from Nature - Ppsn ViI, vol. 3242. Berlin: Springer-Verlag Berlin, 2004, pp. 722-731.
[43] K. S. Bhattacharjee et al., "Bridging the Gap: Many-Objective Optimization and Informed Decision-Making," IEEE Trans. Evol. Comput., vol. 21, no. 5, pp. 813-820, 2017.
[44] L. Rachmawati and D. Srinivasan, "A multi-objective evolutionary algorithm with weighted-sum niching for convergence on knee regions," in GECCO 2006, Seattle, Washington, USA, pp. 749–750, 2006.
[45] K. Zhang et al., "Knee based multimodal multi-objective evolutionary algorithm for decision making," Inf. Sci., vol. 544, pp. 39-55, 2021.
[46] J. Branke et al., "Guidance in evolutionary multi-objective optimization," Adv. Eng. Softw., vol. 32, no. 6, pp. 499-507, 2001.
[47] K. Deb and S. Gupta, "Towards a Link between Knee Solutions and Preferred Solution Methodologies," 2010, pp. 182-189.
[48] S. Sudeng and N. Wattanaongpaksorn, "A knee-based multi-objective evolutionary algorithm: an extension to network system optimization design problem," Cluster Comput., vol. 19, no. 1, pp. 411-425, 2016.
[49] C. Ramirez-Atencia et al., "A knee point based evolutionary multi-objective optimization for mission planning problems," in GECCO 2017, Berlin, Germany, 2017, pp. 1216–1223, 2017.
[50] S. Chouchaicharoenkul and N. Wattanaongpaksorn, "Post Pareto-optimal ranking algorithm for multi-objective optimization using extended angle dominance," Expert Syst. Appl., vol. 158, 2020.
[51] N. Riquelme et al., "Performance metrics in multi-objective optimization," in CLEI 2015, pp. 1-11, 2015.
[52] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: a comparative case study
and the strength Pareto approach," IEEE Trans. Evol. Comput., vol. 3, no. 4, pp. 257-271, 1999.

SUPOJ CHOACHAICHAROENKUL received the B.Sc. degree in computer science from Ramkhamhaeng University (RU) in 2007, B.Eng. degree in electronics engineering in 2011, and M.Eng. degree in electrical engineering from South-east Asia University (SAU) in 2014. He is currently a Ph.D. student in electrical and computer engineering program at King Mongkut’s University of Technology Thonburi (KMUTT). His research interests include multi-objective optimization, evolutionary computation, and orienteering problem.

David Coit is a professor in the Department of Industrial and Systems Engineering at Rutgers University. His research interests are in multi-objective optimization, energy systems optimization and system reliability. His research has been funded by National Science Foundation (NSF), U.S. Navy, U.S. Army and industry. He had over 120 published journal papers and 90 peer-reviewed conference papers. He has a B.Sc. degree in Mechanical Engineering from Cornell University, MBA from Rensselaer Polytechnic Institute, and M.Sc. and Ph.D. degree from University of Pittsburgh. He is a member of IISE, INFORMS and IEEE.

Naruemon Wattanapongsakorn received her Ph.D. in Computer Engineering from University of Pittsburgh, Pennsylvania, USA in 2000. She received her M.S. and B.S. in Computer Engineering from The George Washington University, Washington D.C., USA. Currently, she is an Associate Professor and Ph.D. Program Chair at Computer Engineering Department, King Mongkut’s University of Technology Thonburi, Bangkok, Thailand. Her research areas are multi-objective optimization, computational intelligence, and network security. Dr. Wattanapongsakorn serves as editorial board and reviewer for a number of outstanding international journals, and technical program committee for many IEEE international conferences.