International Journal of Smart Computing and Artificial Intelligence
International Institute of Applied Informatics
2017, Vol.1, No.2, P.113 - 135

Proposal and Progress of a Road-map to Bridge Theoretical and Practical Approaches for Elevator Operation Problems

Tsutomu Inamoto* , Yoshinobu Higami* , Shin-ya Kobayashi*

Abstract

In this paper, the authors propose a road-map to bridge theoretical and practical approaches in the discipline of the elevator operation problem. The theoretical approach aims to solve static elevator operation problems, here *static* denotes all information on users of the elevator system is known before scheduling. The practical approach aims to construct rule-bases for realistic situations, where the user's behavior is not known in detail, but known to obey a certain traffic pattern. The authors expect efforts for bridging those approaches to yield a supervised learning of rule-bases by using optimal solutions as teaching data.

The proposed road-map is comprised of 5 stages: (1) to obtain an optimal solution for a problem instance of a static elevator operation problem, (2) to construct an *identical* optimal rule-base from the optimal solution, (3) to construct a *similar* optimal rule-base which is based on some characteristic functions and effective only for that problem instance, (4) to construct a rule-base which is effective for a set of problem instances, and finally (5) to construct a rule-base which is effective for various problem instances. Computational result display current progress in earlier stages of the road-map.

Keywords: elevator operation problem, integer linear programming, rule-base, simulation

1 Introduction

People in cities use elevator systems (ESs)[1] in their everyday life to move vertically in buildings. Making ESs effective has a direct benefit for their users; we want to reduce waiting times for elevators (cars) as short as possible. An effective ES is also beneficial for bosses of tenants in a building, as shorter waiting times for cars will lead to (slightly) longer working times of employees and may increase their productivity. This kind of significance of the ES is prominent in developing countries, because more and more buildings are constructed day by day. Another kind of significance of the ES can be found in developed countries. Such a country hardly expects the growth

* Ehime University, Matsuyama, Japan

of national population, and less and less buildings are required month by month. But taller and taller buildings will become necessary, because the nation will want people to crowd into cities so as to suppress social costs on infrastructures. A taller building inevitably involves a larger ES. Such a ES has a large degree of freedom in scheduling, so it becomes considerable that the difference between effective and ineffective car operations. Thus the authors think the significance of the research on ESs has widely increased, and will last.

One of the authors has conducted researches to realize effective car operations[2]. Here the effectiveness of a car operation is evaluated as the dissatisfaction of users (passengers) of an ES, as it has been the primary concern of researchers for a long time[3]. Nowadays another concern has rapidly raised from the standpoint of the social cost to sustain a society. That concern aims to suppress power consumption of ESs. However, the authors focus only on the dissatisfaction of passengers, because of such conjecture that car operations with lower power consumption are obtainable by extending the objective function which represents the dissatisfaction of passengers. Thus, in this paper, the elevator operation problem (EOP) is tackled as a problem of obtaining car operations which minimize the dissatisfaction of passengers.

The authors point out that there are 2 approaches in tackling the EOP. The one approach is to simplify the EOP as the static optimization problem, then to solve that problem[2]. This approach will be preferred by academic researchers, because it yields optimal, ideal solutions. But practitioners will not prefer that approach, because it is merely applicable to unreal situations. Practitioners will prefer the other approach of obtaining operations which are essentially applicable to real ESs and are more effective than existing operations. In this paper, those approaches are called theoretical and practical, respectively. These approaches have been proceeded independently in different researches by different researchers. Here, the authors believe that bridging those 2 approaches is fruitful, and devote this paper to develop a way for bridging those approaches.

The one of the authors has developed seminal 2 tools for theoretical and practical approaches. The one tool for the theoretical approach is the trip-based model[4] in formulating static EOPs as integer linear programming (ILP) problems. In this formulation, the ES is assumed to be discretized. By using that tool, we can obtain optimal solutions even for such an innovative ES like multi-car ES [5, 6, 7] in theory. The other tool for the practical approach is the pragmatic rule design (PRD), which has been thought out on the assumption that a considered ES is continuous[8]. The PRD decodes a rule for car allocations as a discrete vector which represents relative characteristics which distinguish effective cars from others. These 2 tools had made the authors to conceive such a fruit to construct rule-bases due not to trial-and-errors but optimal solutions.

Based on aforementioned conjectures and tools, the authors propose a road-map which deploys optimal solutions of static EOPs to construct rule-bases applicable to realistic situations. The proposed road-map is comprised of 5 stages: (1) to obtain an optimal solution for a problem instance of a static EOP, (2) to construct an *identical* optimal rule-base from the optimal solution, (3) to construct a *similar* optimal rule-base which is based on some characteristic functions and effective only for that problem instance, (4) to construct a rule-base which is effective for a set of problem instances, and finally (5) to construct a rule-base which is effective for various problem instances. The objective of this paper is to describe that road-map

and to display current progress on its earlier stages.

This paper is organized as follows. Section 2 is devoted to describe the EOP. In Sections 3 and 4, the trip-based model and the PRD are briefly introduced, respectively. The proposed road-map is described in Section 5. Computational results on current progress of that road-map are displayed in Section 6. Section 7 summarizes this paper and displays future works.

2 Elevator Operation Problem

2.1 Difficulties in Operating Elevator Systems

The elevator system (ES) is said a most familiar transportation system for people to move vertically in buildings. In this paper, the problem of effectively operating ESs is called the elevator operation problem (EOP), and the most typical ES is considered. In such ES, each shaft contains one car, and no *reverse-run* can occur; that is, a car with passengers can not move toward the direction opposite to which intended by passengers.

It is analyzed that the EOP has 3 difficulties[4]. The first difficulty is the indeterminacy of the ES, as an ES can not sense information on passengers thoroughly and predict passengers' behaviors. The second difficulty is that the ES behaves in so complicated manner that the state transition of the ES is hardly represented as mathematical equations. This difficulty is called *curse of modeling*[9]. Even if these difficulties are solved, the EOP is reduced to a combinatorial optimization problem which is too difficult to be solved. For example, the size of the solution space in an EOP with 3 cars and 10 passengers is roughly estimated as $3^{10} \cdot (2 \cdot 10)! \simeq 1.43 \cdot 10^{23}$. This is the third difficulty. These difficulties seem not unique in the EOP but common in many scheduling problems. So researches on these difficulties are expected to have other applications than the EOP.

2.2 Short Overview on the Literature

One of the authors have tackled to the EOP from 2002, and have published a survey paper in Japanese at 2012 [10]. There are some academic studies [11, 12, 13, 14] which have treated the EOP as optimization problems. Especially, the optimal operation rule for the up-peak traffic pattern was displayed in [14], whereas that rule requires such unavailable information as the arrival rate of passengers. Other studies have no guarantee to yield optimal solutions, due to the incomplete formulation [12], the utilization of the receding horizon approach [11], and the utilization of the genetic algorithm [13]. One of the authors made it possible to yield optimal solutions for the deterministic and stochastic EOPs by deploying the integer linear programming [4, 16] and the dynamic programming [15], respectively. However, they are not applicable to real problems, since the problem size becomes exponentially huge on the deterministic, and stochastic EOPs according to the number of passengers and the number of cars, and the number of floors and the number of cars, respectively.

On the other hand, there have been many industrial studies for the EOP. Most of them deployed such technologies as the fuzzy rule, the genetic algorithm, and the agent [17]. Especially the fuzzy rule seems dominant [18, 19, 20], although the number of published papers has decreased. Most of those studies were conducted by

researchers in industries, and destined applicable to real problems. Those studies displayed that their methods had been better than other methods. However, the optimality of their methods was not analyzed, since optimal solutions of real problems are not available.

This contrast between academic and industrial studies appears not only on the legacy ES but also such new ESs as the destination entry ES [17], the double-deck ES [21] and the multi-car ES [6]. The authors envisions to integrate academic and industrial studies owing to the scalability of the PRD which is roughly described at Section 4.2. We can think there are 2 kinds of scalability on a rule-design; weak and strong. The weak scalability means that a rule-base for small-scaled problem instances is almost optimal for middle-scaled problem instances. As like, the strong scalability means that a rule-base for small-scaled problem instances is almost optimal for large-scaled problem instances. We speculates it is possible to display that the PRD has the weak scalability. If such an expectation is valid that a rule design which exhibits the weak scalability also exhibits the strong scalability, then we can obtain near-optimal pragmatic rule-bases for large-scaled EOPs by solving small-scaled EOPs then constructing those rule-bases. It is quite difficult to confirm that expectation. However, the authors think the aforementioned vision has a certain value even when that expectation is invalid, since studies to display the weak scalability of the PRD contribute to the discipline of the EOP.

2.3 Problem Instance

Passengers of an ES form traffic flows due to their activities. Four typical flows are the up-peak, down-peak, two-way, and inter-floor[1]. The up-peak flow is formed by passengers who go to their offices, and usually observed in the morning. The down-peak flow is formed by passengers who return to their homes, and usually observed in the evening. The two-way flow is a mixture of the down-peak and up-peak flows, where the former is formed by passengers who go to lunch and the latter is formed by those who return to their offices. The inter-floor is observed in remainder periods and has no notable features. The combination of a traffic flow and an intensity of passengers' arrivals is called traffic pattern in this paper. A noticeable example of the traffic pattern is the strong up-peak, which is observed in taller buildings with many tenants. It is a necessary condition of an ES to be equipped within a building that the ES can convey passengers in the strong up-peak pattern without intolerable waiting times[3].

Passengers' arrivals are appropriately modeled as a Poisson process, so passengers can be sampled by using pseudo random numbers when a certain traffic pattern is specified. Given data of information on passengers and initial floors of cars, the EOP becomes static in the sense that all information which makes the behavior of the ES indeterministic is known before scheduling. A set of those data is called problem instance in this paper. By focusing on a certain problem instance, the first difficulty at Section 2.1 is dissolved. Static EOP and its optimal solution seem useless, since it is impossible to obtain any problem instance in a real ES by nowadays technologies. However, an optimal solution can be used as a touchstone of a heuristics by being compared with results by that heuristics. Moreover, we can expect that a scheduler which is effective for many problem instances of a certain traffic pattern is also effective for other problem instances sampled from the same traffic

pattern. This paper is essentially based on that expectation.

2.4 Objective Function

For a long time, a "good" car operation for an ES is that minimizes the dissatisfaction of passengers of that ES. Nowadays, such a car operation is also regarded good from economical and sustainable standpoints that minimizes power consumption. Therefore we find 2 objectives in the EOP[4]. The latter objective on power consumption can be basically measured in terms of a number of stops and/or a length of trajectories of cars, so can be integrated into a sole objective function with the former objective. Thus the authors has judged that it is sufficient to consider only the objective function to minimize the dissatisfaction of passengers. The integration of those 2 objectives is out of the scope and one of future works of this paper.

The dissatisfaction of a passenger can be measured in terms of his/her waiting time, traveling time, and long-wait. The waiting time of a passenger is the duration from the moment at when he/she arrives to the moment at when he/she rides a car. As like this measure, the traveling time of a passenger is defined on moments at when he/she rides and leaves from a car. The long-wait represents that a passenger has waited longer than a given threshold Δ. Let denote t_i^{w}, t_i^{t}, and h_i to aforementioned 3 measures respectively with regard to passenger $i \in \mathscr{P} := \{1, \ldots, N^{\mathrm{p}}\}$, here N^{p} denotes the number of passengers who use the ES in a given planning horizon. By denoting r_i, t_i^+, and t_i^- to the arriving time, the riding time, and the leaving time of passenger i, symbols t_i^{w}, t_i^{t}, and h_i are defined as follows:

$$t_i^{\mathrm{w}} := t_i^+ - r_i, \quad t_i^{\mathrm{t}} := t_i^- - t_i^+, \quad h_i := \begin{cases} 1 & \text{if } t_i^{\mathrm{w}} > \Delta, \\ 0 & \text{otherwise .} \end{cases} \tag{1}$$

By using these symbols, the objective function in this paper is defined as follows:

$$\sum_{i \in \mathscr{P}} \left(w^{\mathrm{w}} t_i^{\mathrm{w}} + w^{\mathrm{t}} t_i^{\mathrm{t}} + w^{\mathrm{L}} h_i \right), \tag{2}$$

here $w^{\mathrm{w}}, w^{\mathrm{t}}$, and w^{L} are non-negative scalars which denote the weights of waiting time, traveling time, and long-wait of passengers, respectively. The objective of the static EOP is to obtain car operations which minimize Equation (2).

2.5 Two-index Formulation Model

The static EOP can be handled as a kind of the pickup-and-delivery problem (PDP)[22], which is a special variation of the vehicle routing problem (VRP)[23]. In the PDP, decision variables are classified into 2 types. The one type is assignments of items to vehicles, and the other is sequences of picking-up and delivering items for each vehicle. This model for formulating also applies to the static EOP, thus its fundamental decision variables are assignments of passengers to cars, and processing sequences of passengers for each car.

In the PDP, the processing sequence of items is usually formulated by using binary variables each of which represents the precedence relation between 2 items. This formulation is called 2-index formulation model[23], since each binary variable is indexed by 2 subscripts which correspond to 2 items between which the precedence relation is considered. In the 2-index model, there are n^2 binary variables

if there is n items. Applying this model to the static EOP results into $O(|\mathscr{P}|^2)$ binary variables[16]. They are so numerous that hamper to solve larger problems. Additionally, the 2-index model often brings such an intuitive inconsistency that it requires longer computational times to obtain solutions which result into simpler car trajectories[16]. The one of the authors had conjectured that there must be another formulation that requires shorter computational times when optimal car trajectories are simpler.

3 Trip-based Integer Linear Programming Model for Elevator Operations

3.1 Notion of Trip

It is one of major differences between the EOP and the PDP that they have 1 and 2 dimensions of kinetic planes, respectively. That is, in a usual ES, a car can not move horizontally and repeats only up-and-down movements. This may bring the well-known *reverse-run* constraint; a car should not move toward the direction opposite to the traveling direction of a passenger contained in that car. There must be a proper formulation that deploys this reverse-run constraint to shrink the search space of the EOP. This perspective has been realized by introducing the notion of *trip*[13, 4], which denotes in this paper a one-directional movement of a car and comprises a car trajectory. A sketch of trips is illustrated in Figure 1, in where one car forms a trajectory which is comprised of 2 trips and conveys 3 passengers. This sketch also displays that up- and downward trips are numbered odd and even, respectively. The authors adopt this numbering scheme in formulating the EOP. By using this notation, the reverse-run is easily prohibited. Without the prohibition of the reverse-run, any passenger can be assigned to any trip. With that prohibition, upward/downward passengers can be assigned only to odd/even trips, respectively. This limitation on assignments reduce the search space.

3.2 Trip-based Formulation Model

The processing sequence of passengers assigned to a same trip can be determined in a straightforward manner, since a car repeats up-and-down movements and the reverse-run is prohibited in this paper. For example, passenger i with origin floor f should be picked-up before another passenger i' with origin floor $f' > f$, if they are assigned to a same upward trip. Thus, by grounding the EOP on the notion of trip, we can consider car operations according to assignments of passengers to trips. This fact leads us to the trip-based model[4] for formulating such assignment problems as integer linear programming problems.

In the trip-based model, time, positions of cars, and the behavior of the car are roughly discretized, and all cars are assumed to stay at their initial floors without any passenger at time 0. Decision variables considered in the model are: the first-time, first-floor, last-time, and last-floor of trips, the riding time, leaving time of passengers. They are subsidiary in the sense that, their values are automatically determined due to constraints for prohibiting the reverse-run and connecting trips when the objective function value is minimized. So the trip-based model makes it possible to solve a problem instance with many passengers when fewer trips are

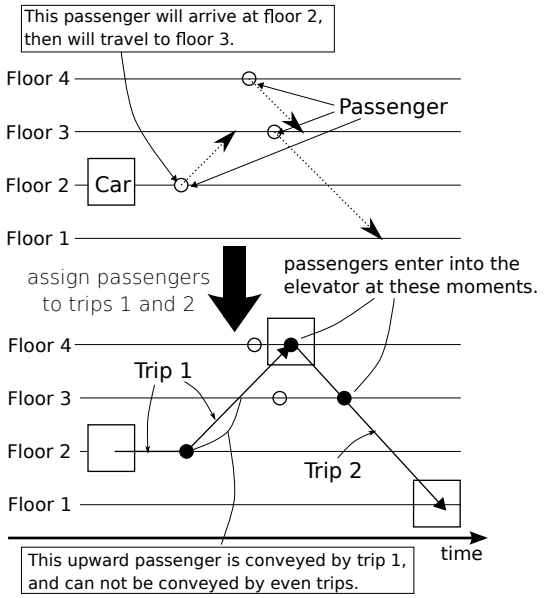

Figure 1: A sketch of trips.

sufficient to comprise optimal trajectories. Currently there is no idea to directly know the number of sufficient trips. That number is left as a parameter specified by a decision-maker. Such a feature of the trip-based model seems a defect that the number of trips is a parameter. This is right when the objective is to obtain the optimal solution. But if the objective is to obtain an effective solution within a limited time, then we can regard that feature as a merit because a solution with a certain effectiveness can be rapidly obtained by solving ILP equations generated with few trips.

In current utilization of the trip-based model, the number of sufficient trips is auxiliary obtained by the incremental procedure[4]. This procedure starts to set that number 2; all trajectories of cars can not contain 3 or more trips. We can obtain ILP equations with 2 trips, then solve it by applying a mathematical solver. If 2 trips are sufficient to represent a feasible solution, we can obtain an optimal solution with 2 trips. As like 2 trips, we can obtain an optimal solution with 4 trips. Since it is pursued to minimize the objective function, the objective function value of the optimal solution with 4 trips is less or equal to that with 2 trips. These 2 values must coincide if enough trips has been considered. The incremental procedure repeats to increase trips and to solve ILP equations until the objective function value converges. Thus, the computational time for solving a certain problem instance by the incremental procedure is a summation of all repetitions over different trip numbers. And the optimal solution of a problem instance is that obtained by solving ILP equations with sufficient trips. This incremental procedure is used in this paper.

3.3 Two Contrivances to Decrease Computational Times

Computational resources to solve a static EOP grow exponentially according to the scale of the problem. Even a small EOP may require a long computational time and a large memory. In order to decrease the computational time, the authors have

proposed 2 contrivances[4, 24].

The one contrivance is relevant to sole car and used in allocating trips to passengers. This contrivance is based on the following 2 conjectures: (i) an optimal car operation is graded in terms of the number of trips in the sense that more trips can bring more elaborated car trajectories, and (ii) an optimal car trajectory of low-graded will be similar to that of slightly higher-graded. These conjectures lead to the procedure which increases but limits the search space by imposing a proximity condition[4]. The procedure firstly solves a 2-graded problem by applying a mathematical solver to ILP equations with 2 trips. Let us assume that we can obtain a 2-graded solution of the 2-graded problem, and 2-graded allocations of trips to passengers. Then the procedure solves a 4-graded problem on the proximity condition that possible allocations of trips in 4-graded problem are not so different from 2-graded allocations. By denoting $l_i^{(n)}$ to the trip allocated to passenger i in n-graded solution, that proximity condition is formulated as follows:

$$\left| l_i^{(n)} - l_i^{(n+2)} \right| \leq \theta, \quad \forall i \in \mathscr{P}. \tag{3}$$

Here, θ denotes the threshold parameter. By imposing this condition, the number of possible values of $l_i^{(n+2)}$ reduces from $\lceil n/2 \rceil + 1$[1] to $\theta + 1$, thus the search space of $(n+2)$-graded problem shrinks especially when n is large. This procedure loses optimality. But it has been displayed that the degree of degradation was not harmful whereas the computational time was drastically decreased, especially when many trips were involved[4].

The other contrivance was devised to cope with multiple cars. This contrivance premises that all cars are indistinguishable. This premise is valid when initial floors of cars are identical and capacities of cars are also. If m cars are indistinguishable, then there are $m!$ optimal solutions which are different in assignments of passengers to cars but identical in the objective function value. This symmetry on car is undesirable in applying a branch-and-bound procedure, since partial solutions which differ only in car allocations have same lower bounds, and none of them is bounded. The contrivance mentioned here breaks that symmetry by giving such artificial roles to cars that a car with a smaller index has a smaller assignment pattern number of passengers (APN)[24]. An APN of a car is a decimal number converted from the binary vector which denotes assignments of passengers to that car. This contrivance may be not so powerful, but has such a desirable property that not to lose optimality.

In this paper, aforementioned 2 contrivances are always adopted in solving ILP equations. So it should be noted that a solution obtained in stage 1 (described in Section 5 lately) is not optimal but quasi optimal. It is one of future works to think out a contrivance which can decrease computational times without losing optimality.

Decision variables of the trip-based ILP formulation are displayed in the appendix. Constraints of the formulation are beyond the scope of this paper, thus omitted. For more details, please refer [4].

[1]The figure 2 in the denominator comes from the "reverse-run" constraint.

4 Elevator Operations by Rule-Bases

4.1 If-then Rule Design

In early days, the ES had contained only one car. The EOP for such ES reduces to a simple problem of scheduling sequences of passengers. For this sequencing, the reverse-run constraint gives birth to the famous selective-collective heuristics (SC); collecting passengers (selectively) with a same direction[1, 11]. The SC is simple but effective[2]. Therefore it is usual in practice that only the car allocations are in question and the passenger sequences are determined by the SC.

Let us imagine to schedule car allocations to passengers according to a rule-base. A typical rule design is the *if-then* design. An if-then rule $\langle C : A \rangle$, where C is a condition part and A is an action part, designates that "when a passenger pushes a button in an ES, car A is allocated to that passenger if the state of the ES matches C."

In order of an if-then rule-base to function well, its members are to be minute. Therefore, the number of possible if-then rule is enormous, and is roughly proportional to the size of the state space of the ES. The ES in itself is familiar with us, but its state space is not. The discretized state space of an ES is huge even if its scale is middle (e.g. 4 cars in a building with 10 floors[8, 9]). The if-then rule space is also huge, which hampers a simple method like the genetics-based machine learning (GBML)[25] to obtain effective rule-bases.

4.2 Pragmatic Rule Design

As described in Section 4.1, the size of the rule space is vital for effective car operations. The first step to decrease the rule space is not to consider direct attributes of cars and passengers, but to use characteristics such as the expected waiting time of a passenger for a certain car. However, this step still requires many rules in representing even simpler heuristics. For example, let us image to represent such heuristics that "to allocate a car with a shortest expected waiting time," and the rule design is $\langle w_1, w_2 : k \rangle$. This design designates that "if expected waiting times of cars 1 and 2 are respectively w_1 and w_2, then allocate car k." If indices of possible cars are $\{0,1\}$ and possible expected waiting times are $\{0,1,2\}$, then the following 6 rules are necessary to completely represent the aforementioned heuristics: $\langle 0,1:0 \rangle$, $\langle 1,0:1 \rangle$, $\langle 0,2:0 \rangle$, $\langle 2,0:1 \rangle$, $\langle 1,2:0 \rangle$, $\langle 2,1:1 \rangle$. As like, the number of rules is summed up to $|\mathscr{W}|^{N^c} \cdot N^c$, where $\mathscr{W}$ is the set of possible expected waiting time and N^c is the number of cars.

The second step for decreasing the rule space is to normalize characteristics between cars then to discretize them, finally to expel the action part from the rule. The resultant rule selects an arbitrary car which has relative characteristics specified by itself. For example, the second step converts the aforementioned heuristics to a compact rule: $\langle 0 \rangle$. This rule selects car 0 in the cases of $\langle w_1, w_2 \rangle \in \{\langle 0,1 \rangle, \langle 0,2 \rangle, \langle 1,2 \rangle\}$, and car 1 in the cases of $\langle w_1, w_2 \rangle \in \{\langle 1,0 \rangle, \langle 2,0 \rangle, \langle 2,1 \rangle\}$. That rule says "any car is OK if it functions well," thus is called pragmatic. So the rule design is called pragmatic rule design (PRD) in this paper. A PRD rule for car operation has such an advantage that its validity is irrelevant to the number of cars[8] and, in probable, the number of floors. This scalability on the number of cars has been confirmed

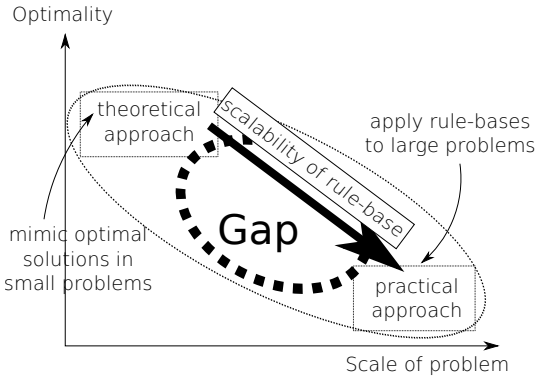

Figure 2: The gap between theoretical and practical approaches, and the image to bridge those approaches

as rule-bases for an ES with 4 cars were also effective for ESs with 8, 12, and 16 cars[8]. When a PRD rule is used as population, the car allocation becomes more adaptive by majority voting[25]. The authors think such scalability makes the PRD a seminal idea for the practical approach.

5 Road-Map to Bridge Theoretical and Practical Approaches

5.1 Motivation

Using the trip-based model makes it possible to obtain (quasi) optimal solutions for problem instances. The approach to obtain (quasi) optimal solutions for mathematically idealized situations is said theoretical. The optimality of those solutions is obvious but it is also that they are not applicable to real situations. On the other hand, applying the GBML makes it possible to obtain effective PRD rule-bases. The approach to obtain applicable solutions for realistic situations is said practical. The applicability of rule-bases is obvious but their optimality is not, as contrary to the theoretical approach. These characteristics of the theoretical and practical approaches indicate a big gap between them as displayed in Figure 2. We can expect some fruits by filling this gap. This expectation leads to the theme of this paper: filling that gap.

In bridging the theoretical and practical approaches, it will be necessary to extend the theoretical approach toward the practical one, and vice versa. The image of extending both approaches is displayed in Figure 2. The authors think that the trip-based model is based on the trip, which is a physical feature of car trajecto-ries, so has the potential to be extended to realistic situations. From the practical approach, the PRD has unlimited potential to represent any heuristics, although the realizability of such potential depends on whether proper characteristics are us-able or not. This means that the practical approach can be extended by devising appropriate characteristic functions, each of which represents a degree of a certain effect caused by selecting a car at a certain state. It is expected that extending both approaches makes such story come true that obtaining (quasi) optimal solutions for limited EOPs, constructing rule-bases by using optimal solutions as teaching data,

then applying resultant rule-bases to real EOPs. In this story, the performance of a constructed rule-base must be worse than that of corresponding optimal solutions. However, this degree of degradation will lessen as the gap between those approaches shrinks by extending them.

The final objective of this research is to bridge the theoretical and practical approaches in the disciple of the EOP. The objective of this paper is to propose a road-map as a bridgehead of that bridge, and to display current achievements in earlier stages of the road-map.

5.2 Five Stages of the Road-Map

We can classify the EOP with regard to the scale and the predictability. The scale is defined on the number of floors in which the considered ES is equipped, the number of cars of that ES, and the number of passengers who use that ES during a given planning horizon. The predictability is categorized into *static* and *dynamic*. *Static* means that arrival times, origin and destination floors of all passengers in the planning horizon are thoroughly known. Complementary situations are *dynamic*. By using these terms, real EOPs are said large in scale and dynamic in predictability. On contrary, EOPs solvable by the trip-based model are said small in scale and static in predictability.

We can classify the car operation rule with regard to the effectiveness, the applicability, and the similarity. The effectiveness directly corresponds to the objective function value. The applicability is categorized into *peculiar*, *narrow*, and *wide*. *Peculiar* means that a rule is only applicable to a certain problem instance. Here, "a rule is applicable" means that rule is not so worse than a typical heuristics. *Narrow* means that a rule is applicable to some problem instances. At last, *wide* means that a rule is applicable to various problem instances, which are usually sampled from a certain traffic pattern. From the standpoint of this paper's theme, those terms are used as: a *narrow* rule-base is constructed by supervised learning with many (quasi) optimal solutions, and that rule-base is expected to be *wide* due to the scalability of the rule design. The similarity is categorized into *identical* and *similar*. *Identical* means that a rule is applied on the basis of unique IDs of passengers and cars, and is valid only for a certain problem instance with *static* predictability. *Similar* means that car operations are determined by using some characteristics, which are computed on the basis of the instant state of the ES.

Based on the aforementioned 3 properties of the car operation rule, the road-map to bridge the theoretical and practical approaches is analyzed and divided into 5 stages as shown in Table 1. The goals of those 5 stages are as follows:

Stage 1 To solve larger static problem instances within shorter computational times.

Stage 2 To develop a rule design which determines car operations based on (quasi) optimal solutions of stage 1.

Stage 3 To develop a rule design which can mimic rules of stage 2 by deploying proper characteristic functions.

Stage 4 To elaborate a rule design of stage 3 in order not to conflict with each other in the sense that an effective rule for a certain problem instance is also effective or inactive for other problem instances.

Table 1: Proposed road-map comprised of 5 stages.

Stage	Predictability	Applicability	Similarity
1	To obtain (quasi) optimal solutions by solving many problem instances of static EOPs.		
2	static	peculiar	identical
3	static	peculiar	similar
4	dynamic	narrow	similar
5	dynamic	wide	similar

Stage 5 To generalize a rule design of stage 4 so as to function well for problem instances which are unseen but obey a same traffic pattern, which is behind problem instances considered in stage 4.

At stages from 2 to 5, car trajectories are produced by using a continuous simulator like the one used in [8]. In this simulator, most properties of elements in an ES take continuous values, and the car accelerates and decelerates according to the specification of that ES. Therefore, the EOP in stage 1 is thoroughly discretized and can be formulated in the ILP, whereas the EOP is continuous from stage 2 and can not be formulated in the ILP. Thus, from stage 2, we can not obtain solutions which are optimal in themselves, but obtain rules which reproduce optimal solutions. The solutions in stage 1 are used as ideal results in stages from 2 to 4. Those solutions are also used in stage 5 to investigate the situation when appropriate results are not obtained. The rules in stage 2 are used as ideal results in stage 3 in the sense that rules in stage 3 are required to produce same results by rules in stage 2. The rules in stage 3 are used as a draft version of rules in stage 4. The rules in stage 3 function well for a certain problem instance, but will not for a set of problem instances. If rules become to function well for a set of problem instances, then those rules are outcomes of stage 4. The rules in stage 4 are used as a draft version of rules in stage 5. The rules in stage 4 function well for a certain set of problem instance, but will not for various sets of problem instances. Here, each set is sampled from the traffic pattern used to sample problem instances in stage 1. If rules become to function well for various sets, then those rules are outcomes of stage 5.

The final goal of our research is to fill the gap between stages 1 and 5. That gap is too wide and can not be filled by a few of studies. Therefore, the road-map is proposed in order to encourage researchers including us to steadily conduct studies focusing on a certain gap between succeeding 2 stages. It should be noted each gap is rather subjective, and a certain gap can be easily filled when a researcher sets its boundaries closer. However, the easiness in filling a certain gap will make it more difficult to fill the next gap. Thus, we think it is better not to pursue a certain stage in serial but to conduct studies on all stages in parallel, thereafter to tackle for filling the entire gap. This parallel approach takes quite long time, but will help us to avoid local optima.

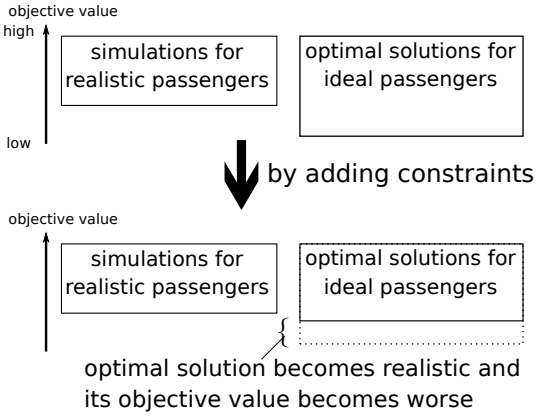

Figure 3: A sketch of the influence by adding constraints for worsening optimal solutions.

5.3 Some Efforts to Fill Gaps

5.3.1 On Gap between Stages 1 and 2

It is necessary to sequentially fill gaps between stages i and $i+1$. The first gap is between stages 1 and 2. This gap may be caused by the following 4 points: (i) the difference on the riding behavior of passengers, (ii) the difference on the irritated behavior of passengers, (iii) the difference on car trajectories, and (iv) the *identical* rule.

As to the point (i), in a continuous simulator (used from stage 2), passenger i with an earlier arrival time is forced to ride ahead of other passenger i', whose traveling direction and origin floor are same to those of passenger i, and who has arrived later than passenger i. This limitation on riding orders of passengers does not exist in stage 1, as that limitation frequently yields non-optimal solutions. A possible way to coincide stages 1 and 2 with regard to the point (i) is to worsen solutions in stage 1 by forcing earlier passengers to be assigned for earlier trips. Here, "worsening solutions" means to add constraints to integer linear programming problems, and to make it impossible to obtain impractical solutions. This image is sketched in Fig. 3. By denoting N^{l} to the number of trips and defining $\mathscr{L} := \{1,\ldots,N^{\mathrm{l}}\}$, that constraint is formulated as follows:

$$\sum_{l\in\mathscr{L}} l x_{k,l}^{(i)} \leq \sum_{l\in\mathscr{L}} l x_{k,l}^{(i')} \quad \text{if } \sum_{l\in\mathscr{L}} x_{k,l}^{(i)} = \sum_{l\in\mathscr{L}} x_{k,l}^{(i')} = 1 \\ \left(\forall i,i' \in \mathscr{P}\left(i\neq i'\right) \middle| r_i \leq r_{i'} \wedge f_i^+ = f_{i'}^+ \wedge d_i = d_{i'}\right). \quad (4)$$

Here, $x_{k,l}^{(i)} \in \{0,1\}$ is a binary variable which denotes passenger i is assigned to trip l of car k or not. Furthermore, f_i^+ and d_i denote the origin floor and traveling direction of passenger i, respectively.

As to the point (ii), passenger i may wait for other passengers forever after he/she has ridden into a car in stage 1, whereas such passenger becomes irritated then pushes close-buttons of doors in stages from 2. A possible way to cope with this difference is to consider the longest time during when an inside passenger waits for others, then to limit the temporal length of a trip by using that longest time.

Table 2: Examined values of T^{m}, T^{c}, and T^{w}.

Parameter	Values
T^{m}	7
T^{c}	1, 3, 5
T^{w}	5, 10, 100
θ	2

By denoting T^{w} to such longest time, and T^{c} to the time consumed in passengers' riding into or leaving from a car, that limitation is forumulated as follows:

$$t^{\mathrm{l}-}_{k,l} \leq t^{\mathrm{l}+}_{k,l} + T^{\mathrm{m}} \left| f^{\mathrm{l}+}_{k,l} - f^{\mathrm{l}-}_{k,l} \right| + N^{\mathrm{p}} (2T^{\mathrm{c}} + T^{\mathrm{w}}) \quad \text{if } \exists i \in \mathscr{P} \left(x^{(i)}_{k,l} = 1 \right) \\ (\forall k \in \{1, \ldots, N^{\mathrm{c}}\}; l \in \mathscr{L}). \quad (5)$$

Here, $t^{\mathrm{l}+}_{k,l}$ and $t^{\mathrm{l}-}_{k,l}$ denote the first and last times of trip l of car k. As like, $f^{\mathrm{l}+}_{k,l}$ and $f^{\mathrm{l}-}_{k,l}$ denote the first and last floors of that trip. T^{m} denotes the reciprocal of the moving speed of a car.

The point (iii) means that the car is modeled to move with a constant speed without acceleration nor deceleration in stage 1, whereas they are accelerated and decelerated in stages from 2. To cope with this difference, parameters in the trip-based model are examined so that car trajectories in stage 1 become similar to those in stage 2. Those examined parameters are T^{m}, T^{c}, and T^{w}. T^{m} is fixed 7 according to an everyday experience, and other parameters are examined as shown in Table 2.

As to the point (iv), it is the most straightforward rule which is comprised of 2 components each of which takes one of $\{0, \ldots, N^{\mathrm{v}}, \#\}$, here N^{v} is such a positive integer that satisfies $N^{\mathrm{v}} \geq \max(N^{\mathrm{p}}, N^{\mathrm{c}}) - 1$. If an *identical* rule is $\langle k, p \rangle$, it means that "'allocating car with ID k to passenger with ID p' is specified in the corresponding (quasi) optimal solution." The *identical* rule-base for an optimal solution is composed of such N^{p} rules, as there are N^{p} passengers. It should be noted that the normalization over cars is not conducted, thus an *identical* rule is not a PRD rule.

The necessary (but not sufficient) condition for accomplishing stage 2 is that objective function values of car trajectories resulted by *identical* rule-bases are always better than those by any heuristics for car operation. Because an *identical* rule-base has to represent a (quasi) optimal solution, which must be better than or equal to all feasible solutions which contain a solution produced by a certain heuristics. Thus, we can say that the gap between stages 1 and 2 is not filled if *identical* rule-bases are sometimes worse than a heuristics in terms of the objective function value.

5.3.2 On Gap between Stages 2 and 3

It is not known that which characteristics are most suitable for the *similar* rule in stage 3. However, it is certain that more and more characteristics a rule has, higher and higher the probability of accomplishing stage 3. Thus it is rational to devise then use as many characteristics as possible. For example, 55 characteristic functions are defined in the aforementioned simulator, and all of them are used in this paper. Their details are omitted due to the space limitation, whereas 12 of them had displayed in [8].

The construction of a *similar* rule-base (in stage 3) is simple. At first, a problem instance is given, then its (quasi) optimal solution is obtained in stage 1. At second, that (quasi) optimal solution is converted to an *identical* rule-base without any consideration on a simulator. Then, the *identical* rule-base is deployed with a simulator like [8]. For each moment when a passenger pushes a hall-call button in simulation, an *identical* rule for that passenger is activated, then we can know its resultant car by the rule and the state of the ES at that moment. By using the resultant car and the ES's state, a vector of N^{s} components is computed by normalizing each of N^{s} characteristic values over N^{c} cars, discritizing them as integers, then selecting N^{s} integers which corresponds to the selected car. Here, N^{s} denotes the number of characteristics in stage 3. Repeating this procedure converts the *identical* rule-base into a *similar* rule-base of N^{p} rules, each of which has N^{s} components.

The necessary and sufficient condition for accomplishing stage 3 is that all car allocations by a *similar* rule-base are equal to those by its ancestral *identical* rule-base. Such concrete verification is not conducted in this paper, and is one of future works.

6 Computational Results

6.1 Parameter Setting

One of 2 objectives of this paper has been accomplished as displayed in Section 5. The remain is devoted to display current achievements in earlier stages of the road-map.

The stage 1 has been partly accomplished by proposing the trip-based formulation model as described in Section 3, and by developing 2 contrivances to decrease computational times as described in Section 3.3. The stages 2 and 3 have been partly accomplished by proposing the PRD as described in Section 4.2, and by describing the *identical* and *similar* rule designs in Section 5. This section examines gaps among those stages in terms of objective function values of 3 groups of car trajectories. Car trajectories in the first and second groups are produced by *identical* and *similar* rule-bases of stages 2 and 3, respectively. Car trajectories in the third group are produced by a heuristic called CDSC. Here the CDSC selects a car which is smallest in the expected waiting time, and is typically used in selecting cars[3]. The specifications of the simulated ES are, although not complete, displayed in Table 3 with parameters of considered traffic patterns. Here, the scale of considered problems is smallest, because the authors think it is more important to thoroughly accomplish 5 stages on small problems than to accomplish only earlier stages on large problems. Other specifications of the simulator are same to [8].

All computations were conducted on a computer with 2 Xeon E5-2640 (v2) 2.0 [GHz] CPUs and a main memory of 64 [GB]. The OS of that computer was CentOS 6.8. IBM CPLEX[26] version 12 was used the mathematical solver to solve ILP problems.

6.2 Procedure to Examine Achievements of Stages 1 to 3

The computation starts from generating some problem instances for each traffic pattern. In stage 1, basically one set of ILP equations is generated for each problem

Table 3: Parameter setting.

Parameter	Value
Number of floors	6
Number of cars (N^{c})	3
Number of passengers (N^{p})	10
Capacity of cars	5 [person]
Number of problem instances	10
Traffic flow	up-peak, down-peak, two-way
Expected arrival interval	5
w^{w}, w^{t}, w^{L}, Δ	1, 1, 60, 60

instance. Other sets are generated and used, if it is intended to examine influence of imposing some constraints. That set results into an (quasi) optimal solution by the mathematical solver. In stage 2, one *identical* rule-base is generated for each (quasi) optimal solution without simulator. The objective function value of that rule-base is calculated from car trajectories produced by simulating the ES used in [8] with car allocations determined by that *identical* rule-base. In this simulation, a *similar* rule-base is generated for each *identical* rule-base, thus stage 2 partially overlaps with stage 3. That *similar* rule-base is evaluated as like *identical* one, and its objective function value is also measured. The degrees of achieving stages 2 and 3 are examined in terms of objective function values as described at the former paragraphs of Sections. 5.3.1 and 5.3.2, respectively.

6.3 Computational Times

Ten problem instances were generated for each traffic pattern, thus 30 problem instances were resulted in total. Four sets of ILP equations were generated for each problem instance. Those sets differed in whether the constraints of Equations (4) and (5) were imposed or not. The computational times of those 4 sets are displayed in Table 4, where each of those computational times is an average over 10 computational times which were required in solving corresponding sets of ILP equations by the mathematical solver. In this table, T^{c} and T^{w} columns display figures of Table 2, and *FCFS* and *length* columns represent whether the constraint of Equations (4) and (5) were imposed on the corresponding ILP equations or not.

Table 4 display that obtaining quasi optimal solutions for the down-peak traffic pattern requires longer computational times than the up-peak traffic pattern. This tendency accords with the fact that devising a heuristics for the up-peak traffic flow is easier than doing so for the down-peak traffic flow[1]. Thus, the computational time in solving ILP equations by the trip-based model for a certain problem instance may have a proportional relation with the hardness in devising an effective heuristics for that instance. If this conjecture is valid then it results that operating an ES with shorter floor heights and/or longer stopping times is rather hard, as Table 4 display larger T^{c} had required longer computational times.

6.4 Objective Function Values

6.4.1 In Stage 2

As like Table 4, objective function values of stage 2 were averaged then displayed in Table 5. Each objective function value was measured from cars' trajectories which were generated by *identical* rule-bases of a certain quasi optimal solution.

Table 5 displays that objective function values with regard to the considered up-peak and down-peak traffic patterns decrease as T^c increases. This indicates that quasi optimal solutions with $T^c \in \{1,3\}$ in stage 1 are far from optimal in stage 2, and setting $T^c = 5$ is important to reduce the gap between stages 1 and 2. When focusing on results with $T^c = 5$, imposing constraints of Equations (4) and (5) has decreased objective function values. This indicates that those constraints make quasi optimal solutions in stage 1 more realistic, thus reduce the gap between stages 1 and 2. However, as to the two-way traffic pattern, we have to say that the gap is still opened, since objective function values in stage 2 are rather larger than those by the CDSC as displayed in the most right column of Table 5.

6.4.2 In Stage 3

In order to examine the achievement level of stage 3, objective function value ratios were calculated then displayed in Table 6. This ratio is of an objective function value in stage 3 to stage 2 with regard to a same problem instance. We can say that the gap between stages 2 and 3 is narrow when that ratio is about 1, since such ratio indicates that car trajectories generated by a *similar* rule-base are alike those generated by its ancestral *identical* rule-base, at least in terms of the objective function value.

In Table 6, we can see that that maximum difference of those ratios from 1 is 0.0848. This value indicates that a *similar* rule-base was deteriorated at the level of 9% at most. This figure is not so small, thus it is improper to state that the gap between stages 2 and 3 is almost filled. Possible countermeasures for narrowing that gap are devising effective constraints and thinking out proper characteristics. In devising such constraints, it may be a cue that those ratios on the up-peak traffic pattern seem far from 1.

7 Conclusion

In this paper, the authors proposed a road-map to bridge theoretical and practical approaches for the elevator operation problem. That road-map is comprised of 5 stages as described in Section 5. Two constraints were proposed to deteriorate solutions of stage 1. Those solutions are more realistic, thus those constraints can reduce gaps between stages 1 and 2. Computational results displayed that those constraints worked except for the two-way traffic pattern on small problems, and there is a wide gap between stages 2 and 3 in terms of objective function value ratios. Thus further efforts are necessary.

Most urgent work is to fill the gap between stages 1 and 2 for the two-way traffic pattern. Unfortunately, there is no cue to attain that work, so it may be necessary such detailed investigation including the comparison of cars' trajectories. Another approach is to use a different ES simulator like [5]. It must be considered also how

to verify achievements of the road-map in an easier manner. Other future works are to integrate the objective on dissatisfaction of passengers and power consumption of elevator systems, and to think out a contrivance which can decrease computational times without losing optimality.

References

[1] G. R. Strakosch, ed., *The Vertical Transportation Handbook*. John Wiley & Sons, Inc., 3 ed., 1998.

[2] T. Inamoto, H. Tamaki, H. Murao, and S. Kitamura, "An application of branch-and-bound method to deterministic optimization model of elevator operation problems," in *Proc. SICE Annual Conference 2002*, 2002, pp. 1345–1350; doi:10.1109/SICE.2002.1195304.

[3] G. Barney, *Elevator Traffic Handbook — Theory and Practice*. Spon Press, 2003.

[4] T. Inamoto, C. Ohta, and H. Tamaki, "Gradually resolving procedures by a trip-based integer programming to optimize elevator operations," in *Proc. The 6th International Conference on Soft Computing and Intelligent Systems & The 13th International Symposium on Advanced Intelligent Systems (SCIS-ISIS-2012)*, 2012, pp. 626–632; doi:10.1109/SCIS-ISIS.2012.6505108.

[5] T. Miyamoto and S. Yamaguchi, "MceSim: A multi-car elevator simulator," *IEICE Transactions on Fundamentals and Electronics*, vol. E91-A, no. 11, 2008, pp. 3207–3214; doi:10.1093/ietfec/e91-a.11.3207.

[6] G. Reuter, "TWIN lift system part two: Technical description," *Elevator World*, vol. 52, no. 4, 2004, pp. 58–64.

[7] A. V. Chian and T. Miyamoto, "Performance evaluation of an option-based learning algorithm in multi-car elevator systems," *IEICE Transactions on Fundamentals and Electronics*, vol. E95-A, no. 4, 2012, pp. 835–839; doi:10.1587/transfun.E95.A.835.

[8] T. Inamoto, C. Ohta, H. Tamaki, and H. Murao, "An approach employing polysemous rules to complement legacy rules for the elevator operation," *Journal of Advanced Mechanical Design, Systems, and Manufacturing*, vol. 4, 2010, pp. 651–663; doi:10.1299/jamdsm.4.651.

[9] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, 1998.

[10] T. Inamoto, "Recent research trends for elevator operations — from an optimization viewpoint," *Systems, Control and Information*, vol. 53, no. 3, 2012, pp. 120–127 (written in Japanese).

[11] S. Tanaka, Y. Uraguchi, and M. Araki, "Dynamic optimization of the operation of single-car elevator systems with destination hall call registration: Part I. formulation and simulations," *European Journal of Operational Research*, vol. 167, 2005, pp. 550–573; doi:10.1016/j.ejor.2004.04.038.

[12] Z. Shen and Q. Zhao, "A branch and bound method to the continuous time model elevator system with full information," in *Proc. SICE Annual Conference 2007*, 2007, pp. 327–330; doi:10.9746/jcmsi.1.467.

[13] J. Sun, Q.-C. Zhao, and P. B. Luh, "Optimization of group elevator scheduling with advance information," *IEEE Trans. Autom. Sci. Eng.*, vol. 7, no. 2, 2010, pp. 352–363; doi:10.1109/TASE.2009.2024242.

[14] D. L. Pepyne and C. G. Cassandras, "Optimal dispatching control for elevator systems during uppeak traffic," *IEEE Trans. Control Syst. Technol.*, vol. 5, 1997, pp. 629–643; doi:10.1109/87.641406.

[15] T. Inamoto, H. Tamaki, and H. Murao, "Model-approximated dynamic programming based on decomposable state transition probabilities," in *Proc. SICE Annual Conference 2007*, 2007, pp. 17–20; doi:10.1109/SICE.2007.4421439.

[16] T. Inamoto, Y. Higami, and S. Kobayashi, "A call-based integer programming model for static elevator operation problems," in *Proc. The 7th International Conference on Soft Computing and Intelligent Systems & The 15th International Symposium on Advanced Intelligent Systems (SCIS-ISIS-2014)*, 2014, pp. 365–369; doi:10.1109/SCIS-ISIS.2014.7044775.

[17] J. Koehler and D. Ottiger, "An AI-based approach to destination control in elevators," *AI Magazine*, vol. 23, no. 3, 2002, pp. 59–78.

[18] M.-L. Siikonen, "Elevator group control with artificial intelligence," tech. rep., Helsinki University of Technology, 1997.

[19] C. B. Kim, K. A. Seong, H. Lee-Kwang, J. O. Kim, and Y. B. Lim, "Fuzzy approach to elevator group control system," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 25, no. 6, 1995, pp. 985–990; doi:10.1109/21.384263.

[20] T. Beielstein, C.-P. Ewald, and S. Markon, "Optimal elevator group control by evolution strategies," in *Proc. the 2003 international conference on Genetic and evolutionary computation*, vol. 2, 2003, pp. 1963–1974; doi:10.1007/3-540-45110-2_95.

[21] J. Sorsa, M.-L. Siikonen, and H. Ehtamo, "Optimal control of double-deck elevator group using genetic algorithm," *International Transactions in Operational Research*, vol. 10, 2003, pp. 103–114; doi:10.1111/1475-3995.00397.

[22] G. Berbeglia, J.-F. Cordeau, I. Gribkovskaia, and G. Laporte, "Static pickup and delivery problems: a classification scheme and survey," *Top*, vol. 15, no. 1, 2007, pp. 1–31; doi:10.1007/s11750-007-0009-0.

[23] P. Toth and D. Vigo, eds., *Vehicle Routing: Problems, Methods, and Applications, Second Edition.* Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 2001.

[24] T. Inamoto, Y. Higami, and S. Kobayashi, "Giving formal roles to elevators for breaking symmetry in static elevator operation problems," in *Proc.IEEE 4th Global Conference on Consumer Electronics (GCCE-2015)*, 2015, pp. 27–30; doi:10.1109/GCCE.2015.7398534.

[25] A. Fernández, S. García, J. Luengo, E. Bernadó-Mansilla, and F. Herrera, "Genetics-based machine learning for rule induction: State of the art, taxonomy, and comparative study," *IEEE Trans. Evol. Comput.*, vol. 14, no. 6, 2010, pp. 913–941; doi:doi:10.1109/TEVC.2009.2039140.

[26] IBM, *IBM ILOG CPLEX Optimizer*.

A Decision variables of the Trip-based ILP formulation

This appendix introduces the decision variables of the trip-based ILP formulation for EOPs in order to indicate the scale of the search space. Only the primary variables are designated to be scheduled. Other subsidiary variables are determined according to the values of the primary variables and constraints which connect trips and so on. For more details, please refer the original work[4].

- Primary variables:
  - $x_{k,l}^{(i)} \in \{1,0\}$ $(i \in \mathscr{P};\ k \in \mathscr{C};\ l \in \mathscr{L})$: Denoting whether passenger i is assigned to trip l of car k or not. Here, $\mathscr{C}$ denotes the set of car indices.
- Subsidiary variables for passenger $i \in \mathscr{P}$:
  - $t_i^+,\ t_i^-$: Denoting the times when the passenger enters into and leaves from a car, respectively.
  - $h_i \in \{1,0\}$: Denoting the passenger waits longer than a given threshold Δ or not.
- Subsidiary variables for trip $l \in \mathscr{L}$ of car $k \in \mathscr{C}$:
  - $t_{k,l}^{\mathrm{l}+},\ t_{k,l}^{\mathrm{l}-}$: Denoting the starting and finishing times of that trip, respectively.
  - $f_{k,l}^{\mathrm{l}+},\ f_{k,l}^{\mathrm{l}-}$: Denoting the starting and finishing floors of that trip, respectively.

Table 4: Computational times in obtaining quasi optimal solutions.

T^{c}	T^{w}	FCFS	length	Objective function value		
				up-peak	down-peak	two-way
1	5	no	no	6.446	37.725	15.806
1	5	no	yes	9.59	42.987	11.477
1	5	yes	no	6.036	27.29	9.237
1	5	yes	yes	5.203	30.67	11.263
1	10	no	no	5.809	38.297	10.938
1	10	no	yes	5.57	26.367	12.828
1	10	yes	no	5.191	30.871	11.027
1	10	yes	yes	6.77	34.681	11.348
1	100	no	no	6.937	31.031	9.529
1	100	no	yes	7.755	48.887	11.923
1	100	yes	no	6.105	31.154	8.644
1	100	yes	yes	5.518	29.387	14.504
3	5	no	no	10.3	64.803	27.583
3	5	no	yes	14.637	67.565	31.415
3	5	yes	no	13.801	42.961	18.577
3	5	yes	yes	12.575	50.893	17.968
3	10	no	no	12.268	48.684	27.189
3	10	no	yes	12.972	70.529	24.588
3	10	yes	no	14.445	31.307	19.878
3	10	yes	yes	23.639	51.872	24.461
3	100	no	no	9.943	48.761	28.5
3	100	no	yes	16.29	53.146	37.961
3	100	yes	no	13.612	31.527	18.579
3	100	yes	yes	15.004	42.926	24.572
5	5	no	no	45.747	99.616	39.987
5	5	no	yes	48.554	104.81	41.007
5	5	yes	no	27.656	118.32	30.181
5	5	yes	yes	23.86	119.84	47.565
5	10	no	no	47.612	98.844	40.126
5	10	no	yes	41.49	139.32	42.878
5	10	yes	no	21.767	118.85	37.419
5	10	yes	yes	36.982	118.05	42.79
5	100	no	no	40.508	99.288	38.833
5	100	no	yes	52.668	140.95	35.722
5	100	yes	no	20.041	117.84	30.163
5	100	yes	yes	39.285	149.43	34.788

Table 5: Objective function values in stage 2.

T^c	T^w	FCFS	length	Objective function value		
				up-peak	down-peak	two-way
1	5	no	no	466.1	596.3	545.4
1	5	no	yes	470.9	596.6	563.2
1	5	yes	no	472.3	591.5	556
1	5	yes	yes	477.1	601.4	562.5
1	10	no	no	466.1	596.3	545.4
1	10	no	yes	442.5	596.6	545.4
1	10	yes	no	472.3	591.5	556
1	10	yes	yes	470.9	544.7	545.4
1	100	no	no	466.1	596.3	545.4
1	100	no	yes	477.1	544.7	549.4
1	100	yes	no	472.3	591.5	556
1	100	yes	yes	466.1	544.7	549.4
3	5	no	no	476.9	586.3	596.4
3	5	no	yes	461.1	584.1	577.6
3	5	yes	no	466.9	593.8	619
3	5	yes	yes	469.2	591.6	592.2
3	10	no	no	476.9	586.3	596.4
3	10	no	yes	469.6	591.2	611.6
3	10	yes	no	466.9	593.8	619
3	10	yes	yes	445.3	586.7	604.9
3	100	no	no	476.9	586.3	596.4
3	100	no	yes	450.2	584.1	603.6
3	100	yes	no	466.9	593.8	619
3	100	yes	yes	439.6	584.5	604.9
5	5	no	no	438.2	553.3	575.2
5	5	no	yes	434.8	526.5	577.8
5	5	yes	no	436.3	569.8	567.6
5	5	yes	yes	427.1	530.3	569.3
5	10	no	no	438.2	553.3	575.2
5	10	no	yes	445.5	527.5	555.9
5	10	yes	no	436.3	569.8	567.6
5	10	yes	yes	433.2	553.3	574.2
5	100	no	no	438.2	553.3	575.2
5	100	no	yes	447.4	527.5	561.7
5	100	yes	no	436.3	569.8	567.6
5	100	yes	yes	423.2	527.5	563.9
(CDSC)				501.49	550.46	529.99

Table 6: Objective function value ratios of stage 3 to stage 2.

T^{c}	T^{w}	FCFS	length	Objective function value		
				up-peak	down-peak	two-way
1	5	no	no	0.94551	0.99681	1.0062
1	5	no	yes	0.94606	0.98827	0.97532
1	5	yes	no	0.95702	0.99679	1.0061
1	5	yes	yes	0.95745	0.98836	1.006
1	10	no	no	0.94551	0.99681	1.0062
1	10	no	yes	0.97469	0.98827	1.0062
1	10	yes	no	0.95702	0.99679	1.0061
1	10	yes	yes	0.94606	0.99651	1.0062
1	100	no	no	0.94551	0.99681	1.0062
1	100	no	yes	0.95745	0.99651	1.0062
1	100	yes	no	0.95702	0.99679	1.0061
1	100	yes	yes	0.94551	0.99651	1.0062
3	5	no	no	1.0824	1.0086	1
3	5	no	yes	1.0309	1.0086	1
3	5	yes	no	1.0653	1.0085	1
3	5	yes	yes	1.0811	1.0085	1
3	10	no	no	1.0824	1.0086	1
3	10	no	yes	1.0771	1.0085	1
3	10	yes	no	1.0653	1.0085	1
3	10	yes	yes	1.0858	1.0086	1
3	100	no	no	1.0824	1.0086	1
3	100	no	yes	1.0412	1.0086	1
3	100	yes	no	1.0653	1.0085	1
3	100	yes	yes	1.0844	1.0086	1
5	5	no	no	1.0396	1.0091	1.0072
5	5	no	yes	1.0399	1.0096	1.0071
5	5	yes	no	1.0523	1.0089	1.0073
5	5	yes	yes	1.0145	1.0095	1.0073
5	10	no	no	1.0396	1.0091	1.0072
5	10	no	yes	1.0512	1.0096	1.0074
5	10	yes	no	1.0523	1.0089	1.0073
5	10	yes	yes	1.0401	1.0091	1.0072
5	100	no	no	1.0396	1.0091	1.0072
5	100	no	yes	1.051	1.0096	1.0074
5	100	yes	no	1.0523	1.0089	1.0073
5	100	yes	yes	1.0411	1.0096	1.0073