


The Elevator Dispatching Problem

Jari Ylinen

Related papers

[Download a PDF Pack](#) of the best related papers 



[A Survey of Elevator Group Control Systems for Vertical Transportation: A Look at Recent Lit...](#)
Pablo Cortés

[Dynamic Fuzzy Logic Elevator Group Control System for Energy Optimization](#)
Pablo Cortés

[Design of Fuzzy Based Controller for Modern Elevator Group with Floor Priority Constraints](#)
S.M. Hasanul Banna Kasemi

The Elevator Dispatching Problem

Janne S. Sorsa

KONE Corporation, Janne.Sorsa@kone.com

Harri Ehtamo

Helsinki University of Technology, Harri.Ehtamo@hut.fi

Marja-Liisa Siikonen, Tapio Tyni, Jari Ylinen

KONE Corporation, Marja-Liisa.Siikonen@kone.com, Tapio.Tyni@kone.com, Jari.Ylinen@kone.com

In this paper, we formulate the elevator dispatching problem as a stochastic bilevel optimization problem, where a group of elevators serves a set of transportation requests. In the upper level, the requests are assigned to the elevators. In the lower level, the route for each elevator is solved separately. The lower level problem predicts the optimal system state for a finite planning horizon according to stochastic demand. The demand affects the cost of traveling, capacity constraints, and the upper level payoff function. To solve the problem in real time, we use a genetic algorithm to optimize the upper level assignments. In the lower level optimization, we use the certainty equivalent control scheme, where stochastic quantities are replaced by their expected values. Extensive numerical simulations carried out in this paper, together with commercial installations, verify the applicability of our new on-line optimization algorithm.

Key words: vehicle routing; stochastic programming; certainty equivalent controller

History: Submitted: September 2009.

1. Introduction

In the Elevator Dispatching Problem, EDP, a group of elevators serves a set of transportation requests. The EDP is a stochastic and dynamic pickup and delivery problem, where future requests are uncertain at the instant of making dispatch decisions. It is also dynamic in a strict sense as defined in Psaraftis (1988), since the problem inputs may change while either executing the algorithm or the elevators follow the planned routes. In this paper, we present a mathematical formulation of EDP, the algorithm to solve it, as well as an on-line implementation of the optimal dispatch decisions. Thus, our work adds an important contribution and a real world application to the field.

Traditionally, elevator dispatching has dealt with simple questions such as to which transportation requests the elevators are dispatched next. We depart from this tradition by building up complete elevator routes up to the point where all transportation requests are served. Moreover, we minimize the total cost defined by the elevator routes, which results in optimal dispatch decisions. New EDP instances are solved, and new optimal decisions generated in an on-line fashion either

at predefined instants or as soon as new requests have appeared. According to current elevator control requirements, each problem instance has to be solved in less than half a second.

In this paper, we give a mathematical formulation of EDP with stochastic demand and unknown destinations. The purpose of our formulation is to provide a general framework that allows modeling of passenger-related uncertainties. Passengers press up or down call buttons to register pickup requests to their desired traveling direction in the lobbies of their departure floors. The number of waiting passengers, i.e., stochastic demand in terms of Gendreau et al. (1996), is uncertain since the arrivals are assumed to follow a Poisson process (Alexandris 1977). The destination floors of the passengers are unknown until they enter the elevator and press the destination buttons to register delivery requests. At the solution instant of the problem, however, the exact number of passengers traveling to a particular destination floor is not known.

The EDP can be considered as an assignment problem, which in addition to ordinary constraints contains a set of traveling salesman problems, TSP, as constraints. In the assignment problem, we first assign transportation requests to elevators. Each elevator then visits the assigned requests in a tour loop fashion. This kind of optimization problem is called hierarchical or bilevel optimization problem (Shimizu et al. 1997). We apply a well-known solution procedure for such a problem. For every upper level iteration giving a set of assignments, we solve a lower level TSP for each elevator.

We formulate lower level TSPs as stochastic optimal control problems, where we consider the stochastic demand. Owing to the uncertainties, EDP has stochastic service times, as defined in Laporte et al. (1992), as well as time-dependent service costs, as in Tagmouti et al. (2007). To solve the TSPs, we apply the idea of the certainty equivalent controller, which is obtained when in the underlying optimization problem the uncertain quantities are replaced by their expectations. Finally, to solve the EDP reliably in real time, we solve the problem in an on-line fashion; see Bertsekas (2005).

A fast solution algorithm is crucial when optimizing elevator routes in a real-time control. In addition, the resulting elevator routes must obey the commonly accepted rules in serving transportation requests, such as Closs rules, Closs (1970). We describe in detail both the upper level assignment algorithm, where a genetic algorithm generates the assignments, and a heuristic algorithm to solve the lower level TSPs. The former was developed by Tyni and Ylinen (2001), but they did not provide explicit formulation of the EDP. The latter is the traditional collective control algorithm, see, e.g., Closs (1970) or Barney (2003). It has not been published earlier in mathematical form. In addition, we here modify it to take into account the stochastic demand, which requires mathematical formulation.

We show by numerical experiments that our algorithm, GA, solves EDP instances fast. The worst-case computation times of GA stay well below the given limit of 500 milliseconds even for the most complex EDP instances, which are rare in practice. Using simulations, we derive regression equations to predict both quantity and quality of service of an elevator group. Kavounas (1993) and Barney (2003) have derived similar equations for predicting quality of service but, to our knowledge, no work exists on predicting the quantity of service. Using the regression equations, we also show that optimization-based GA clearly out-performs the rule-based algorithm called Enhanced Spacing Principle (Siikonen 1997).

The rest of this paper is organized as follows: in Section 2, we review appropriate literature on stochastic and dynamic vehicle routing as well as on elevator dispatching. We formulate EDP in Section 3. In Section 4, we describe our algorithm to solve it and show that elevator routes provided by the algorithm satisfy the Closs rules. Computational experiments are reported in Section 5. Finally, we conclude this paper in Section 6 and discuss directions for future research.

2. Literature Review

Typically, stochastic and dynamic vehicle routing problems are solved in two stages. Initial routes are defined by *a priori* optimization as in, e.g., Jaillet (1988), Bertsimas et al. (1990) and Laporte et al. (1994). During the recourse stage, the routes are adapted to the actual demand, see, e.g., Laporte and Louveaux (1993) and Gendreau et al. (1995). Our real-time control scheme follows this principle. However, each time a new system state is measured, the whole EDP is also solved for the new state. The new solution may define a reassignment of the transportation requests, which is characteristic of the dynamic problem (Psaraftis 1988).

Reassignment of a particular transportation request is allowed until an elevator starts to decelerate to serve it. Thus, the elevator may be diverted from its current destination if the optimal routes change, as discussed in Ichoua et al. (2000). However, diversion affects only the on-line implementation. We apply special techniques in a genetic algorithm to avoid unnecessary diversion arising from the nature of the algorithm, see Section 4.1. The multi-objective method of Sörensen (2006) reduces diversion by measuring the difference in distance between the previous solution and the new solution and uses this distance as an additional objective. Such method could also be implemented in the EDP context.

Dynamic problems also require fast solution algorithms for on-line implementation (Psaraftis 1988). Owing to the on-line requirement and computational complexity, usually heuristic algorithms are used. Bertsimas and van Ryzin (1991) studied several heuristic policies to solve a dynamic

vehicle routing problem and found that the nearest neighbor policy performed well in both light and heavy traffic. Swihart and Papastavrou (1999) came to the same conclusion for the dynamic pickup and delivery problem. A corresponding, well-known policy in elevator dispatching is *collective control*, where the elevator serves the nearest pickup request in front of it (Closs 1970). He also showed that the collective control is nearly optimal for a single elevator, which makes it naturally a good choice as a heuristic for the lower level TSPs.

In old relay-based elevator controls, only simple dispatching rules were available, such as the collective control. Advances in computing technology have determined the development of more complex algorithms for dispatching a group of elevators. Fuzzy logic became popular along with microprocessors (Hirasawa et al. 1978). Later, neural networks were developed to adapt fuzzy rules to the real people flow in buildings (Markon et al. 1994, 2006). A modern elevator control is capable of counting passengers moving in and out of the elevators, recognizing typical traffic patterns, and utilizing this information when dispatching elevators (Siikonen 1997).

However, optimization did not become a feasible alternative until the late 1990s along with the computational power of industrial PCs and the development of system software. Tyni and Ylinen (2001, 2006) developed the first optimization-based elevator control using the Genetic Algorithm, GA, which has been used in KONE AltaTM elevator installations ever since. Sorsa et al. (2003) developed the algorithm further for double-deck elevators based on the ideas of Tyni and Ylinen (2001). Double-deck elevators consist of two elevator cars stacked one on top of the other, moving as single units, and serving adjacent floors simultaneously.

These studies concentrate on EDP *with unknown destinations*, which are related to the traditional user interface of up and down buttons in the lobby. More recently, Tanaka et al. (2005) formulated the routing of a single elevator as a dynamic programming problem, which corresponds to our lower level TSP. They assumed EDP *with known destinations*, where the users input their destination floors already in the lobby. However, they did not consider the stochastic elements at all.

Attempts to combine stochastic information and optimization in EDP have been rare. The earliest one, and the most comprehensive to date, is Levy et al. (1977), in which the optimal control law for an elevator group is derived. However, they assumed that all expected passengers fit in the elevators, which cannot be guaranteed *a priori*. In addition, they did not include passenger transfers in their formulation. This assumption is valid for normal traffic but not if the elevator group is under heavy demand. Our formulation and algorithm overcome these problems and ensure that our approach is valid in all possible traffic situations.

Pepyne and Cassandras (1997) used dynamic programming to derive a threshold-based optimal control policy for morning up-peak traffic. According to their policy, an elevator is dispatched from the lobby when the number of passengers inside it exceeds the threshold. Recently, Wesselowski and Cassandras (2006) introduced a receding horizon control, which decides whether an elevator continues its travel or stops to a particular floor. Because of this control mechanism, their approach requires accurate timing of the decision making in a real-time control. In our control, the elevators travel to their designated floors independent of the control for reliability reasons unless otherwise decided later on. Thus, our approach is robust in this sense for undesirable error situations of real-time controls.

As discussed in Psaraftis (1988), the time factor is essential in dynamic vehicle routing problems. The elevator arrival time to a transport request plays the central role in EDP since it defines, e.g., passenger waiting time. In addition, stochastic demand depends on the arrival time, since passengers are assumed to arrive according to a Poisson process. The stochastic demand, in turn, defines stochastic service time. However, there are only a few earlier studies, where stochastic service times and time-dependency have been considered. These concepts are usually studied separately but in our approach both are included.

Laporte et al. (1992) introduced stochastic travel and service time with independent and known distributions. In EDP, the elevator travel time is deterministic but takes into account acceleration and deceleration (Roschier and Kaakinen 1979). This differs from the usual assumption of constant speed in vehicle routing literature, although Ichoua et al. (2003) studied time-dependent travel times. In EDP, the service time depends on the time it takes for the passengers to enter or exit the elevator. On the other hand, the expected number of waiting passengers increases monotonically with time. Thus, service time is not only stochastic but also time-dependent.

Time-dependent cost usually means the varying cost of traversing an arc at different times, e.g., Picard and Queyranne (1978) and Miller et al. (1994). Psaraftis and Tsitsiklis (1993) use random arc costs depending on environmental variables, which are subject to Markovian transitions. Tagmouti et al. (2007) defined service costs as a function of the time of beginning the service, which is also the case for typical objectives in EDP such as passenger waiting and journey time. In addition, these objectives are also stochastic through the definition of the stochastic demand.

3. Problem Formulation

We state the EDP with the following assumptions. Let us define a complete directed graph $\mathcal{G} = (\mathcal{N}, \mathcal{A})$, where \mathcal{N} is the set of nodes and \mathcal{A} is the set of arcs $(i, j) \in \mathcal{A}$, $i, j \in \mathcal{N}$. Each arc (i, j)

is associated with elevator travel time. The set of nodes \mathcal{N} is further divided into pickup nodes \mathcal{N}^+ , delivery nodes \mathcal{N}^- , and terminal nodes \mathcal{N}^0 . Thus, $\mathcal{N} = \mathcal{N}^+ \cup \mathcal{N}^- \cup \mathcal{N}^0$. The terminal nodes represent either initial elevator positions or expected reversal floors, which are defined dynamically during the solution.

Without loss of generalization, we assume a set of identical elevators $\mathcal{E} = \{1, \dots, E\}$ with respect to *capacity*, *nominal speed*, *acceleration* and *jerk*. Note that E also denotes the number of elevators. The elevators $e \in \mathcal{E}$ are dispatched to serve transportation requests $\mathcal{N}^+ \cup \mathcal{N}^-$. Our aim is to design elevator routes with minimum cost, such that:

- each elevator serves one route, which consists of *trips* in the up and down direction;
- a trip starts and ends at a terminal node;
- each transportation request is visited once by exactly one elevator;
- the capacity of the elevators is not exceeded at any time;
- if overload is detected the elevator bypasses a pickup node and returns to it later on;
- delivery nodes must not be bypassed (Closs 1970).

The problem is inherently dynamic since passengers register new transportation requests over time, to which the elevators are dispatched in real time. In the following, we propose a bilevel formulation for this problem. The upper level assignment problem is formulated as an integer programming problem. For the lower level TSPs, a dynamic programming formulation is given, which takes into account the dynamic and stochastic elements of the problem. The underlying system is continuous in time. However, we assume a discrete-time model, where the time intervals are variable and correspond to elevator travel time. The following additional assumptions are used:

- the elevators \mathcal{E} serve a set of floors $\mathcal{Z} = \{0, \dots, Z\}$ called the zone;
- elevator $e \in \mathcal{E}$ is committed to pickup node $i \in \mathcal{N}^+$ when it starts to decelerate to floor $\zeta(i) \in \mathcal{Z}$;
- the arrivals of the passengers are independent and follow a known probability distribution;
- each pickup node $i \in \mathcal{N}^+$ is associated with a distinct arrival rate $\lambda(i)$.

The late commitment is also known as the *continuous assignment* policy. Another possibility is the *immediate assignment* policy, where an elevator is committed to a pickup request immediately after its registration. The policy affects only the on-line implementation but neither the formulation nor the algorithm. Our implementation follows the continuous assignment policy.

Next, in Section 3.1, the formal model for the upper level problem is given. The lower level problem is formulated in Section 3.2. We defer the explicit definition of upper level payoff function until Section 3.3 because the payoff is expressed in terms of the lower level variables.

3.1. Upper Level Assignment Problem

We first define the decision variables for the upper level problem and then formulate it.

Decision variables. The elevator group \mathcal{E} shares the same set of pickup nodes \mathcal{N}^+ . Without loss of generality, we assume that each pickup node $i \in \mathcal{N}^+$ can be assigned to each elevator $e \in \mathcal{E}$. Let us define the assignment vector $x_e = (x_{ei})$ of elevator $e \in \mathcal{E}$, where x_{ei} is a binary decision variable equalling 1 if pickup node $i \in \mathcal{N}^+$ is assigned to elevator e , and 0 otherwise. The subset $\mathcal{N}_e^+ \subset \mathcal{N}^+$ consists of assigned pickup nodes of elevator e , $\mathcal{N}_e^+ = \{i \in \mathcal{N}^+ | x_{ei} = 1\}$, $\forall e \in \mathcal{E}$. In addition, each elevator has its own delivery and terminal nodes, \mathcal{N}_e^- and \mathcal{N}_e^0 , respectively. Finally, $\mathcal{N}_e = \mathcal{N}_e^+ \cup \mathcal{N}_e^- \cup \mathcal{N}_e^0$.

Assignment problem. The assignment problem is formulated as

$$\begin{aligned} \min_{x_e} \quad & \sum_{e=1}^E F_e(x_e, \bar{\pi}_e(x_e)) \\ \text{s.t.} \quad & \sum_{e=1}^E x_{ei} = 1, \quad \forall i \in \mathcal{N}^+, \\ & x_{ei} \in \{0, 1\}, \quad \forall e \in \mathcal{E}, \forall i \in \mathcal{N}^+, \\ & \bar{\pi}_e(x_e) \text{ solves optimally } P(x_e), \forall e \in \mathcal{E}, \end{aligned} \tag{3.1}$$

where $P(x_e)$ is the lower level TSP, which is defined next in Section 3.2 for given assignments. The assignments x_e defined by an iteration of the upper level problem are parameters to the lower level problem; see section 4.1. The lower level problem then defines the optimal route $\bar{\pi}_e(x_e)$ for the given assignments.

3.2. Lower Level TSP as a Certainty Equivalent Control Problem

We formulate E lower level TSPs, $P(x_e)$, for each elevator $e \in \mathcal{E}$ and the given assignment vector x_e from the upper level. The lower level problem is formulated as a certainty equivalent optimal control problem, where the uncertain quantities are replaced by their expected values. The key idea of our formulation is to connect dynamic and stochastic information in an optimal way and use it in the upper level payoff function $F_e(\cdot)$, which evaluates the quality of the given assignment. The formulation allows flexible definition of constraints, non-linear state equations and cost functions.

Stages. A route of elevator e through the nodes $\mathcal{N}_e = \mathcal{N}_e^+ \cup \mathcal{N}_e^- \cup \mathcal{N}_e^0$ is indexed by stages $k = 0, \dots, K_e$, where $K_e = |\mathcal{N}_e| - 1$. Hence, stages k index the tour of cities in the TSP. Transition from stage k to stage $k+1$ corresponds to elevator e traveling along arc $(u_{e,k-1}, u_{e,k}) \in \mathcal{A}$, where $u_{e,k}, u_{e,k-1} \in \mathcal{N}_e$ is the *control variable*, i.e., a decision variable for the TSP, called so to differentiate it from the upper level decision variable.

Control variables. The control variable of elevator e at stage k , $u_{e,k} \in \mathcal{U}_{e,k}$, determines the node elevator e will visit at stage $k+1$. At each stage, the set of *admissible controls*, $\mathcal{U}_{e,k}$, is restricted

to the nodes in the same direction as the current traveling direction of elevator e . Hence, $\mathcal{U}_{e,k}$ explicitly depends on the *system state*, which is defined below. Also note that $\mathcal{U}_{e,k} \subset \mathcal{N}_e$. If the elevator becomes full, only delivery or terminal nodes are admissible. The set $\mathcal{U}_{e,k}$ will be specified explicitly in Section 4.2, where the algorithm to solve the TSP is defined.

State variables and equations. Let $y_{e,k}$ denote the vector of state variables of elevator e at stage k . Respectively, let the combined state equation be denoted by $y_{e,k+1} = f_{e,k}(y_{e,k}, u_{e,k}, w_{e,k})$, where the random variable $w_{e,k}$ as well as the components of $y_{e,k}$ and $f_{e,k}$ are defined below.

Reversing the current *elevator travel direction* $d_{e,k} \in \{-1, 1\}$ implies the start of a new trip in the opposite direction. The travel direction $d_{e,k+1}$ at stage $k+1$ changes if the only admissible control in direction $d_{e,k}$ is the terminal node,

$$d_{e,k+1} = \begin{cases} -d_{e,k}, & \text{if } |\mathcal{U}_{e,k}| = 1, \\ d_{e,k}, & \text{otherwise.} \end{cases} \quad (3.2)$$

The load of the elevator e , the *number of passengers* inside it, is denoted by $q_{e,k} = 0, \dots, Q$, and updated by the state equation

$$q_{e,k+1} = q_{e,k} + w_{e,k}, \quad (3.3)$$

where $w_{e,k} > 0$ denotes the passengers entering the elevator at stage $k+1$, and $w_{e,k} < 0$ denotes the exiting passengers, respectively. This random variable will be defined explicitly below.

Elevator service on a floor has three phases: the doors open, passengers either enter or exit the elevator, and the doors close. Door *opening* and *closing times* are assumed constants, t_o and t_c , respectively. However, the time elapsing for the passenger transfers is uncertain and depends on the number of transferring passengers, $w_{e,k}$. The *stop time* of elevator e at stage $k+1$ is given by

$$s_{e,k+1} = t_o + t_p |w_{e,k}| + t_c, \quad (3.4)$$

where t_p is the constant *transfer time* of one passenger.

Thus, the state $y_{e,k}$ is defined by $y_{e,k} = (d_{e,k}, q_{e,k}, s_{e,k})$. The components of the function $f_{e,k}$ and the state equations are defined by (3.2), (3.3) and (3.4).

Elevator arrival time. Elevator *travel time* denotes the time for a complete elevator cycle from start at floor $\zeta(u_{e,k-1}) \in \mathcal{Z}$ to start at floor $\zeta(u_{e,k}) \in \mathcal{Z}$. It comprises the uncertain stop time $s_{e,k}$ at stage k and *run time* $\tau(u_{e,k-1}, u_{e,k})$ from $\zeta(u_{e,k-1})$ to $\zeta(u_{e,k})$. Thus, the cost of traveling along arc $(u_{e,k-1}, u_{e,k}) \in \mathcal{A}$ equals $s_{e,k} + \tau(u_{e,k-1}, u_{e,k})$. The run time includes the time to accelerate, to travel at nominal speed, and to decelerate back to rest (Roschier and Kaakinen 1979). Note that run time $\tau: \mathcal{N} \times \mathcal{N} \rightarrow \mathbb{R}$ satisfies the usual assumptions of a distance.

Our aim is to route elevators through the assigned transportation requests in the shortest possible time. Thus, the natural objective of the TSP is the *arrival time* t_{e,K_e-1} of elevator e at the last terminal node $u_{e,K_e-1} \in \mathcal{N}_e^0$ of its route. The arrival time $t_{e,0}$ for the initial stage is given by special rules, which are discussed in Section 4.2. For $k = 1, \dots, K_e - 1$, the elevator arrival time is updated by

$$t_{e,k} = t_{e,k-1} + s_{e,k} + \tau(u_{e,k-1}, u_{e,k}). \quad (3.5)$$

This recursive equation for the arrival time $t_{e,k}$ plays the central role in our formulation since it is the main variable in upper level payoff function $F_e(\cdot)$, which is defined in Section 3.3.

Disturbances. Random disturbances $w_{e,k}$, $k = 0, \dots, K_e - 1$, are characterized by a probability measure $p_{e,k}(\cdot | y_{e,k}, u_{e,k})$ and $\bar{w}_{e,k}$ denotes the expectation of $w_{e,k}$ with respect to $p_{e,k}$. The disturbance $w_{e,k}$ at stage k is the *number of passengers* entering or exiting elevator e in node $u_{e,k} \in \mathcal{N}_e$. Note that in the following, we assume fractional disturbance. The probability distribution $p_{e,k}$ depends on whether $u_{e,k}$ is pickup, delivery or terminal node, as explained below.

For a pickup node $u_{e,k} \in \mathcal{N}_e^+$, the number of passenger arrivals $N(t)$ that have occurred up to time $t \geq 0$ are Poisson-distributed with probabilities $P\{N(t) = n | \lambda(u_{e,k})\}$ for $n = 0, 1, 2, \dots$ passengers, as in, e.g., Larson and Odoni (2007). Without loss of generality, we assume that arrival rates $\lambda(u_{e,k})$, which are needed to define $p_{e,k}$, are available in the elevator control for all $u_{e,k} \in \mathcal{N}_e^+$, and $\lambda(u_{e,k}) > 0$ persons per second. Note that the arrival rate of one person per second equals 300 persons in five minutes and can be regarded as intense traffic for a particular pickup node $u_{e,k} \in \mathcal{N}_e^+$.

In addition, $\gamma(u_{e,k})$ denotes the *call time* elapsed since a passenger registered the pickup request $u_{e,k} \in \mathcal{N}_e^+$. Hence, the total time until elevator e serves $u_{e,k}$ is the call time added to the elevator arrival time, equalling $\gamma(u_{e,k}) + t_{e,k}$. We assume that there is at least one waiting passenger for each pickup node. Then, the probability distribution for pickup node $u_{e,k} \in \mathcal{N}_e^+$ becomes

$$p_{e,k}(w_{e,k} = 0 | y_{e,k}, u_{e,k}) = 0, \quad (3.6)$$

$$p_{e,k}(w_{e,k} = n | y_{e,k}, u_{e,k}) = P\{N(\gamma(u_{e,k}) + t_{e,k}) = n - 1 | \lambda(u_{e,k})\}, n = 1, 2, \dots, \quad (3.7)$$

and the expectation of $w_{e,k}$ is given by

$$\bar{w}_{e,k}(y_{e,k}, u_{e,k}) = 1 + \lambda(u_{e,k})(\gamma(u_{e,k}) + t_{e,k}). \quad (3.8)$$

The destinations of passengers inside the elevator are known at the initial stage and are the given delivery nodes $i \in \mathcal{N}_e^-$. Nevertheless, it is not known, *how many of them* will exit at a particular

delivery node. For simplicity, we assume equal probability for the destination floors of passengers. Then, the expected number of passengers exiting the elevator at delivery node $u_{e,k} \in \mathcal{N}_e^-$ is simply

$$\bar{w}_{e,k}(y_{e,k}, u_{e,k}) = -q_{e,0} / |\mathcal{N}_e^-|, \quad (3.9)$$

where $q_{e,0}$ is the number of passengers inside elevator e at the initial stage.

Similarly, the destinations of passengers entering the elevator at pickup nodes are unknown. These passengers are assumed to exit elevator e at terminal nodes $u_{e,k} \in \mathcal{N}_e^0$. The current load $q_{e,k}$ corresponds to the number of passengers picked up since all passengers initially in the elevator, if any, have already been dropped off at delivery nodes. Thus, for $u_{e,k} \in \mathcal{N}_e^0$,

$$\bar{w}_{e,k}(y_{e,k}, u_{e,k}) = -q_{e,k}. \quad (3.10)$$

Note that the dependence of $\bar{w}_{e,k}$ of the state variables comes only through $q_{e,k}$; and for $u_{e,k} \in \mathcal{N}_e^+$, $\bar{w}_{e,k}$ does not even depend on these. Recall also that $w_{e,k} > 0$ for entering passengers and $w_{e,k} < 0$ for the exiting passengers.

Certainty equivalent control scheme. The EDP instance defined by the assignment problem (3.1) is solved anew in every half a second using a new measurement of \mathcal{N} , and the system state, $\bar{y}_{e,0}$. We seek an 'on-line' implementable algorithm for our dynamic and stochastic EDP problem because the algorithm should give an efficient elevator control scheme in real time. Therefore, in the lower level optimization problem the uncertain elements are replaced by their expected values. Hence, the uncertainties are compensated by a classical deterministic feedback mechanism. We will now define the lower level TSPs as certainty equivalent control problems to be implemented in an 'on-line' fashion. For detailed discussion of such control schemes see Bertsekas (2005, Chap. 6).

Let \mathcal{N} be given, x_e be the current assignment of the upper level iteration (see Section 4.1), and let the measured system state be $\bar{y}_{e,0}$. For all $e \in \mathcal{E}$, we define $P(x_e)$ by

$$\begin{aligned} \min \quad & t_{e,K_e-1} \\ \text{s.t.} \quad & y_{e,k+1} = f_{e,k}(y_{e,k}, u_{e,k}, \bar{w}_{e,k}(y_{e,k}, u_{e,k})), \\ & q_{e,k} \leq Q, \\ & u_{e,k} \in \mathcal{U}_{e,k} \subset \mathcal{N}_e, \\ & k = 0, 1, \dots, K_e - 1, \end{aligned} \quad (3.11)$$

where Q denotes the capacity of an elevator. Above, we have replaced the disturbances $w_{e,k}$ by their expectations $\bar{w}_{e,k}$ with respect to the probability distribution $p_{e,k}$. Let route $\bar{\pi}_e(x_e) = (\bar{u}_{e,0}, \dots, \bar{u}_{e,K_e-1})(x_e)$ be the solution of (3.11) for $e \in \mathcal{E}$. Recall that $\mathcal{N}_e = \mathcal{N}_e^+ \cup \mathcal{N}_e^- \cup \mathcal{N}_e^0$ and that \mathcal{N}_e^+ is the set of all those $i \in \mathcal{N}^+$ for which $x_{ei} = 1$. Hence, the lower level problems and their solutions $\bar{\pi}_e(x_e)$ depend on x_e through the definition of \mathcal{N}_e .

Now, suppose for a while that \mathcal{N} , or \mathcal{N}_e , and x_e are fixed, but we measure the current system state and implement the optimal 'feedback solution' of (3.11) in an on-line fashion. This can be done as follows. First, apply the control input $\bar{u}_{e,0}$, $e \in \mathcal{E}$; then measure the new system state $\bar{y}_{e,1}$, and solve the deterministic control problem (3.11) for stages $k = 1, 2, \dots, K_e - 1$. Let the solution be $\tilde{\pi}_e(x_e) = (\tilde{u}_{e,1}, \dots, \tilde{u}_{e,K_e-1})(x_e)$. Then, apply $\tilde{u}_{e,1}$, $e \in \mathcal{E}$, measure the system state, and repeat the procedure. Note that the subsequent problems can be indexed with respect to stages anew so that $k = 0, 1, \dots, K_e - m$, $m \geq 1$.

Now, during the solution of EDP, x_e in the upper level iteration changes, and hence K_e above also changes until the optimal x_e , with given \mathcal{N} , has been found. Note, however, that for given \mathcal{N} , the optimal x_e , and hence K_e , also remain constant during the on-line implementation of the lower level problem as described above, which includes the measurement of the system state. This implementation ends after the K_e measurements of the system state followed by the K_e applications of new control inputs. The on-line implementation of the whole EDP algorithm defined in Section 4 requires that also the set \mathcal{N} is allowed to change. At every time EDP is solved, the measured \mathcal{N} is also taken into account. If it changes also the optimal x_e changes, which may lead to the reassignment of the transportation requests.

3.3. Upper Level Payoff Functions

The payoff for the upper level problem (3.1) is a sum of E elevators. The payoff function for elevator e is of the following form,

$$F_e(x_e, \bar{\pi}_e(x_e)) = \sum_{k=0}^{K_e-1} g_{e,k}(y_{e,k}, \bar{u}_{e,k}, \bar{w}_{e,k}(y_{e,k}, \bar{u}_{e,k})), \quad (3.12)$$

where $y_{e,k}$ is the system state corresponding to the optimal solution of the certainty equivalent control problem (3.11). Note that $F_e(\cdot)$ as defined by (3.12) also explicitly depends on x_e because $K_e = |\mathcal{N}_e^+ \cup \mathcal{N}_e^- \cup \mathcal{N}_e^0| - 1$. The cost per stage, $g_{e,k}$, can be, e.g., *call time* of elevator,

$$g_{e,k}^{CT} = \gamma(\bar{u}_{e,k}) + t_{e,k}, \text{ if } \bar{u}_{e,k} \in \mathcal{N}_e^+, \quad (3.13)$$

passenger *waiting time*,

$$g_{e,k}^{WT} = (\gamma(\bar{u}_{e,k}) + t_{e,k}) \bar{w}_{e,k}(y_{e,k}, \bar{u}_{e,k}), \text{ if } \bar{u}_{e,k} \in \mathcal{N}_e^+, \quad (3.14)$$

or passenger *journey time*,

$$g_{e,k}^{JT} = \begin{cases} \gamma(\bar{u}_{e,k}) \bar{w}_{e,k}(y_{e,k}, \bar{u}_{e,k}), & \text{if } \bar{u}_{e,k} \in \mathcal{N}_e^+, \\ -(\gamma(\bar{u}_{e,k}) + t_{e,k}) \bar{w}_{e,k}(y_{e,k}, \bar{u}_{e,k}), & \text{if } \bar{u}_{e,k} \in \mathcal{N}_e^-, \\ -t_{e,k} \bar{w}_{e,k}(y_{e,k}, \bar{u}_{e,k}), & \text{if } \bar{u}_{e,k} \in \mathcal{N}_e^0. \end{cases} \quad (3.15)$$

The journey time (3.15) is clear for the passengers inside the elevator, i.e., for $\bar{u}_{e,k} \in \mathcal{N}_e^-$, where $\gamma(\bar{u}_{e,k})$ denotes the time since registering the delivery request. For the passengers waiting for pickup, the journey time consists of the waiting time, $\gamma(\bar{u}_{e,\ell}) + t_{e,\ell}$, $\bar{u}_{e,\ell} \in \mathcal{N}_e^+$, and the travel time inside the elevator, $t_{e,k} - t_{e,\ell}$, $\bar{u}_{e,k} \in \mathcal{N}_e^0$. In addition, $\bar{w}_{e,k}(y_{e,k}, \bar{u}_{e,k})$, $\bar{u}_{e,k} \in \mathcal{N}_e^0$ is the number of passengers that are expected to enter the elevator at the pickup nodes before the terminal node. Above, the minus sign is used because $\bar{w}_{e,k}(y_{e,k}, \bar{u}_{e,k}) < 0$, $\bar{u}_{e,k} \in \mathcal{N}_e^- \cup \mathcal{N}_e^0$. Note also that $\bar{w}_{e,k}(y_{e,k}, \bar{u}_{e,k})$, $\bar{u}_{e,k} \in \mathcal{N}_e^+$ increases monotonically with time and takes into account the passengers that are expected to arrive in the lobby within time $\gamma(\bar{u}_{e,k}) + t_{e,k}$, recall equation (3.8).

4. Numerical Solution Algorithm

In the following, we describe in detail how an EDP instance is solved. First, the genetic algorithm (Goldberg 1989), assigns elevators to $|\mathcal{N}^+|$ pickup requests, as in Tyni and Ylinen (2001). For each assignment given by GA, we solve E lower level TSPs (3.11) and evaluate the fitness of them by using one of the payoff functions (3.13), (3.14), or (3.15). We start by defining the genetic algorithm for the upper level problem.

4.1. Solving Upper Level Problem by Genetic Algorithm

A chromosome \mathcal{C} represents an assignment where each pickup node $i \in \mathcal{N}^+$ is associated with a gene $G \in \mathcal{C}$. Each chromosome defines a solution for the EDP. Note that $|\mathcal{C}| = |\mathcal{N}^+|$. The value of gene G indicates the elevator assigned to pickup node i . The GA maintains a set of chromosomes, population \mathcal{P} , which evolves through generations according to genetic operators. The search space \mathcal{S} defines all the feasible gene combinations for chromosomes. Thus, every assignment \mathcal{C} drawn from search space \mathcal{S} is a feasible solution for an EDP instance. The size of the search space grows exponentially to the number of pickup nodes, $|\mathcal{S}| = E^{|\mathcal{N}^+|}$.

A subset of chromosomes, $\mathcal{B} \subset \mathcal{P}$, having the best fitness values, forms the set of parents. One chromosome of the first generation is initialized with the optimal solution of the previous EDP instance (Tyni and Ylinen 2001). This technique speeds up the search and eliminates unnecessary diversion caused by the random search. In the computation, we use GeneBank developed by Tyni and Ylinen (1999), which can speed up the computation as much as 65 %. The GeneBank stores chromosomes \mathcal{C} and their fitness values $F_{\mathcal{C}}$ to be used directly instead of solving lower level TSPs if the same chromosome reappears in the population. The number of chromosomes in the GeneBank is large but much smaller than the theoretical maximum size of the search space, $|\mathcal{S}|$. Thus, its memory requirement is within the available technology.

Genetic Algorithm

INPUT: Nodes \mathcal{N} and elevators \mathcal{E} . The optimal solution of the previous EDP instance.

OUTPUT: Assignment vectors x_e and route $\bar{\pi}_e(x_e)$ for all $e \in \mathcal{E}$ minimizing $\sum_{e=1}^E F_e(\cdot)$.

STEP 0: INITIALIZE. Set generation = 1. Initialize \mathcal{P} randomly except one chromosome with the solution of the previous EDP instance.

STEP 1: EVALUATE FITNESS. For all $\mathcal{C} \in \mathcal{P}$:

STEP 1.1: Set $x_{ei} = 1$ for $i \in \mathcal{N}^+$, if the value of gene G corresponding to i equals e ; otherwise, set $x_{ei} = 0$.

STEP 1.2: Solve $P(x_e)$ for all $e \in \mathcal{E}$.

STEP 1.3: Evaluate fitness $F_{\mathcal{C}} = \sum_{e=1}^E F_e(x_e, \bar{\pi}_e(x_e))$.

Do

STEP 2: CREATE NEW GENERATION. Set generation = generation + 1. Select parents \mathcal{B} . Apply crossover operator on $\mathcal{C} \in \mathcal{B}$. Apply mutation on children $\mathcal{C} \in \mathcal{P} \setminus \mathcal{B}$.

STEP 3: EVALUATE FITNESS. For all $\mathcal{C} \in \mathcal{P}$, search \mathcal{C} from the GeneBank. If found, set fitness accordingly, otherwise repeat steps 1.1-1.3 for \mathcal{C} and write it and its fitness $F_{\mathcal{C}}$ to the GeneBank.

WHILE generation < max generations; and not all \mathcal{C} of current \mathcal{P} found from the GeneBank.

END OF GENETIC ALGORITHM

Next, we define the algorithm for the lower level problem, which determines the visiting sequence for the assignments given by the genetic algorithm in the upper level.

4.2. Heuristic Algorithm for the Lower Level Problem

The key idea of the algorithm is to select the nearest node to the elevator with respect to its direction of travel. This heuristic rule is assumed to minimize the arrival time at the final stage, t_{e,K_e-1} , and solve the lower level problem efficiently. This principle is also known as the collective control and shown to be nearly optimal (Closs 1970). The rather complex formulation in Section 3.2 is motivated by the need for accurate predictions of arrival time $t_{e,k}$ and disturbance $\bar{w}_{e,k}$ used in the upper level payoff functions. In addition, the formulation allows derivation of exact optimality conditions, which we shall discuss briefly in Section 4.3.

The initial value of the system state, $\bar{y}_{e,0}$, for elevator e is defined by direction $d_{e,0}$ and load $q_{e,0}$ together with stop time, $s_{e,0} = 0$. Direction $d_{e,0}$ and load $q_{e,0}$ are measured from the system. The assigned pickup nodes \mathcal{N}_e^+ are given by the upper level iteration but delivery nodes \mathcal{N}_e^- are measured. If an elevator is standing vacant, a special direction variable is included in the upper level iteration. The value of this variable defines the initial direction $d_{e,0}$.

A terminal node, denoted by $\bar{u}_{e,-1} \in \mathcal{N}_e^0$ to initialize the algorithm, is defined to represent the initial position of the elevator. If elevator e is moving, $\zeta(\bar{u}_{e,-1}) \in \mathcal{Z}$ corresponds to its *advance*

position, i.e., the nearest floor where it can still decelerate. Otherwise, $\zeta(\bar{u}_{e,-1})$ is the floor where it is standing. Direction of $\bar{u}_{e,-1}$ is equal to the initial traveling direction of the elevator, $\delta(\bar{u}_{e,-1}) = d_{e,0}$.

Recall that at each stage $k = 0, \dots, K_e - 1$, the set of admissible controls for elevator e is $\mathcal{U}_{e,k} \subset \mathcal{N}_e$. The optimal control $\bar{u}_{e,k}$ at stage k determines the next node elevator e will visit at stage $k + 1$:

$$\bar{u}_{e,k} = \arg \min_{i \in \mathcal{U}_{e,k}} \tau(\bar{u}_{e,k-1}, i), \quad (4.1)$$

where $\tau(\bar{u}_{e,k-1}, i)$ is the run time of elevator e from floor $\zeta(\bar{u}_{e,k-1}) \in \mathcal{Z}$ to floor $\zeta(i) \in \mathcal{Z}$ along arc $(\bar{u}_{e,k-1}, i) \in \mathcal{A}$. If argmin is not unique, i.e., both a pickup and a delivery node are located on the same floor and in the same direction, then the delivery node is selected first.

For stage $k = 0$ with the optimal control $\bar{u}_{e,0}$, the following rules define the arrival time $t_{e,0}$. If the elevator is moving, $t_{e,0}$ corresponds to the *remaining run time* to the floor of node $\bar{u}_{e,0}$. On the other hand, if the elevator is standing, then $t_{e,0} = 0$. The number of passengers that have already entered or exited the elevator can also be taken into account if necessary.

The admissible controls are nodes towards which elevator e is moving, and which are ahead of it, or on the same floor as it is located. If the elevator is fully loaded at stage k , i.e., $q_{e,k} \geq Q$, only delivery or terminal nodes are admissible. We now explicitly define the set of admissible controls $\mathcal{U}_{e,k}$ of elevator e at stage k ,

$$\mathcal{U}_{e,k} = \{i \in \mathcal{N}_e \setminus \mathcal{V}_e \mid \delta(i) = d_{e,k}, \delta(i)\zeta(i) \geq d_{e,k}\zeta(\bar{u}_{e,k-1})\}, \quad (4.2)$$

where \mathcal{V}_e is the set of visited nodes, which initially contains only terminal node $\bar{u}_{e,-1}$. Execution of the algorithm ends when all nodes have been visited, $\mathcal{N}_e \setminus \mathcal{V}_e = \emptyset$.

Algorithm to solve $P(x_e)$ for $e \in \mathcal{E}$

INPUT: Nodes $i \in \mathcal{N}_e$ as well as floor $\zeta(i) \in \mathcal{Z}$, direction $\delta(i)$, call time $\gamma(i)$ and arrival rate $\lambda(i)$ associated with them, and the measured system state $\bar{y}_{e,0}$.

OUTPUT: Route $\bar{\pi}_e = (\bar{u}_{e,0}, \dots, \bar{u}_{e,K_e-1})(x_e)$ and its payoff $F_e(x_e, \bar{\pi}_e(x_e))$.

STEP 0: INITIALIZE. Set $k = 0$. Measure $y_{e,0}$. Define initial node $\bar{u}_{e,-1} \in \mathcal{N}_e^0$. Set terminal node $i \in \mathcal{N}_e^0$ to correspond to the furthest floor in the direction of travel $d_{e,0}$. Set $\mathcal{V}_e = \{\bar{u}_{e,-1}\}$.

Do

STEP 1: ADMISSIBLE CONTROLS. Set $\mathcal{U}_{e,k}$ as in (4.2).

STEP 2: OPTIMAL CONTROL. Find all $i \in \mathcal{U}_{e,k}$, which minimize elevator run time $\tau(\bar{u}_{e,k-1}, i)$. If there is more than one transportation request node satisfying the argmin condition, then select $i \in \mathcal{N}_e^-$. Set $\bar{u}_{e,k} = i$.

STEP 3: UPDATE. Calculate $t_{e,k}$, $\bar{w}_{e,k}$ and update $F_e := F_e + g_{e,k}$. Set $y_{e,k+1} = f_{e,k}(\cdot)$ as defined by state equations (3.2), (3.3) and (3.4). If $d_{e,k+1} \neq d_{e,k}$, and $\mathcal{N}_e \setminus \mathcal{V}_e \neq \emptyset$, define a new terminal node $i \in \mathcal{N}_e^0$, which is located on the furthest floor in the new direction $d_{e,k+1}$, i.e., $\zeta(i) = 0$ if $d_{e,k+1} = -1$, or $\zeta(i) = Z$ if $d_{e,k+1} = 1$. Add $\{\bar{u}_{e,k}\}$ to \mathcal{V}_e . Set $k = k + 1$.

WHILE $\mathcal{N}_e \setminus \mathcal{V}_e \neq \emptyset$.

END TSP-ALGORITHM

Next, we discuss some properties of the routes defined by our TSP algorithm.

4.3. Closs Rules and Analysis of Lower Level Problem

In this section, we show first that the routes satisfy the statement on good elevator behavior as defined by Closs (1970). Furthermore, we briefly discuss the optimality of the routes.

THEOREM 1. *Let $\bar{u}_{e,k}$ satisfy the argmin condition (4.1). The route $\bar{\pi}_e(x_e)$ satisfies the Closs rules, Closs (1970):*

- (1) *An elevator may not stop at a floor where no passenger enters or leaves it.*
- (2) *An elevator may not pass a floor at which a passenger wishes to leave it.*
- (3) *A passenger may not enter an elevator carrying passengers and travelling in the reverse direction to her required direction of travel.*
- (4) *An elevator may not reverse its direction of travel while carrying passengers.*

Proof of Theorem 1. The first rule is trivial, since \mathcal{N}_e contains only registered pickup and delivery requests for elevator e and terminal nodes are not visited in reality. The second rule is satisfied since the argmin principle selects always the nearest pickup or delivery request ahead of elevator e with respect to its direction of travel, which is initially set towards the delivery requests. The third rule assumes certain behavior of the passengers, which is enforced by separately handling delivery requests in one direction and pickup requests in the other. Since elevator e unloads its passengers either at the delivery nodes or at the terminal node, it is empty before reversing its direction of travel. Thus, also the fourth rule is satisfied. \square

REMARK 1. The third and fourth rules above suggest that the passengers are rational in that they follow the signalization in elevator lobbies. In case the passengers do not behave as expected, the elevator control follows its internal logic.

THEOREM 2. *Let $\bar{u}_{e,k}$ satisfy the argmin condition (4.1). The route $\bar{\pi}_e(x_e)$ solves optimally the lower level problem $P(x_e)$, except for cases considered unusual in practice.*

Proof of Theorem 2 requires that first all possible traffic situations are recognized and then the optimal route is compared to the route defined by our argmin hypothesis. Such situations occur, for

example, when elevator e has no delivery requests, all assigned pickup requests are in one direction, and elevator e does not become full after picking up the pickup requests. This analysis, although straightforward to carry out, is far too long to be included here but will be dealt with in detail in a subsequent paper. In the following, we give a brief outline of the analysis.

For each specific traffic situation we construct two routes, route A using the argmin hypothesis, and another route B not using the argmin hypothesis in all parts of the route. The global optimality of the argmin hypothesis then leads to an inequality whose validity is analyzed with respect to the parameters in question. It turns out that the inequalities for various cases essentially differ only with respect to one case-specific parameter, the magnitude of which determines the optimality of the solution during light traffic, $\lambda(i) \rightarrow 0$ for all $i \in \mathcal{N}_e^+$. In a worst-case situation, the parameter itself implies non-optimality. In addition to this parameter, optimality depends on the expected disturbance, i.e., the number of passengers being picked up and delivered during the route. If disturbance on route A is much greater than on route B then the argmin condition (4.1) may define a non-optimal route.

4.4. Numerical Example of the Heuristic Algorithm

Next, we give a numerical example, which demonstrates how the heuristic algorithm solves the lower level problem. The situation is depicted in Figure 1, which shows a diagram of elevator shaft, transportation requests and passengers on the left, and the resulting graph $\mathcal{G} = (\mathcal{N}, \mathcal{A})$ on the right. The triangles pointing upwards and downwards denote pickup requests and the circles denote delivery requests or expected reversals on terminal floors. For simplicity, the nodes of \mathcal{G} are indexed according to the optimal route, and only the arcs defining this route are shown. Each arc is associated with the cost of traveling along it, i.e., stop time plus run time, $s_{e,k} + \tau(\bar{u}_{e,k-1}, \bar{u}_{e,k})$.

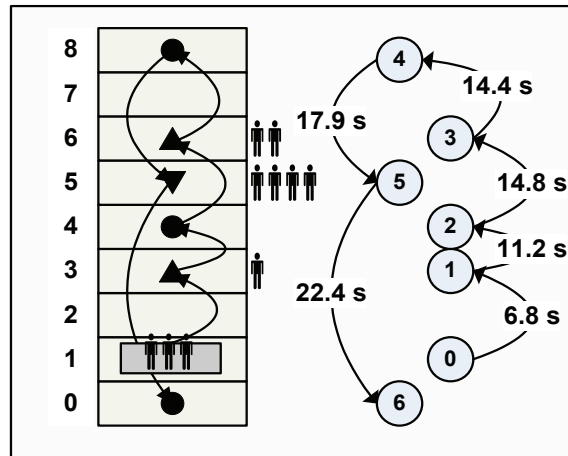


Figure 1 Example of lower level TSP with three pickup requests and one delivery request.

Numerical details of the algorithm are shown in Table 1 below. The terminal node to initialize the algorithm is $\bar{u}_{e,-1} = 0$, which is located on floor $\zeta(0) = 1$. At stage $k = 0$, the optimal control $\bar{u}_{e,0} = 1$ is selected, with floor $\zeta(1) = 3$, and direction $\delta(1) = 1$. At stage $k = 1$, elevator e visits the fourth floor and delivers the three passengers inside it. Then it proceeds to collect passengers on floor 6. The next node on the route is the terminal node, where the passengers from floors 3 and 6 are dropped. After reversing its direction at stage $k = 4$, the elevator picks up passengers from floor 5, which are expected to travel to the first floor.

Table 1 Values of state variables during algorithm execution, $\lambda(\bar{u}_{e,k})$ given in persons per 5 minutes.

k	$d_{e,k}$	$q_{e,k}$	$s_{e,k}$	$\mathcal{U}_{e,k}$	$\bar{u}_{e,k}$	$\zeta(\bar{u}_{e,k})$	$\gamma(\bar{u}_{e,k})$	$\lambda(\bar{u}_{e,k})$	$\tau(\bar{u}_{e,k-1}, \bar{u}_{e,k})$	$t_{e,k}$	$\bar{w}_{e,k}$	$g_{e,k}^{CT}$	$g_{e,k}^{WT}$	$g_{e,k}^{JT}$
0	1	3.0	0	{1, 2, 3, 4}	1	3	12	7	6.8	6.8	1.4	18.8	26.3	16.8
1	1	4.4	6.4	{2, 3, 4}	2	4	0	0	4.8	18.0	-3.0	-	-	53.9
2	1	1.4	8.0	{3, 4}	3	6	20	9	6.8	32.8	2.6	52.8	137.2	52.0
3	1	4.0	7.6	{4}	4	8	0	0	6.8	47.2	-4.0	-	-	188.6
4	-1	0.0	9.0	{5, 6}	5	5	28	11	8.9	65.1	4.4	93.0	409.3	123.2
5	-1	4.4	9.4	{6}	6	0	0	0	13.0	87.4	-4.4	-	-	384.5
6	1	0.0	9.4	{}	-	-	-	-	-	-	-	-	-	-
$F_e(x_e, \bar{\pi}_e(x_e))$												164.6	572.8	819.1

Thus, the optimal route defines that the elevator is dispatched to pickup new passengers waiting on floor 3. In reality, the passengers entering the elevator will register new delivery requests to their real destinations. However, the algorithm does not know these destinations beforehand. As the system state is measured after the stop on floor 3, the new delivery requests are taken into account in the new optimal route defined by the new state.

Note that there is another route for the above situation, which is still feasible with respect to Closs rules but violates our argmin principle. Namely, the passengers in the elevator are first delivered to floor 4, then the pickup request downwards on floor 5 is collected, and only after reversing its direction on floor 0, does the elevator collect pickup requests upwards. It is a straightforward task to verify that this route is worse than the optimal route shown in Figure 1 with respect to the payoff.

5. Simulation Results

In this section, we report the computational experiments, which are conducted using a laptop PC with a 2 GHz processor. In the studied implementation of EDP, total expected passenger waiting time is minimized, recall equation (3.14). In GA, the size of population is 100 chromosomes, from

which the 30 best chromosomes form the group of parents. GA iterates at most 200 generations using uniform crossover with the best parents. Mutation is performed only for children with exactly the same genes as one of its parent, with probability 0.01.

We simulate various traffic situations using Building Traffic Simulator (Siikonen et al. 2001), KONE BTSTM, for 20 elevator groups, see Appendix B. First, we show that our algorithm, GA, solves EDP instances very fast. Next, we derive regression equations from the simulation output of all 20 cases to predict both the *quantity* and *quality* of service of an elevator group. These equations are then used to show that GA outperforms the rule-based *Enhanced Spacing Principle*, ESP, studied earlier by Siikonen (1997). We start by introducing our simulation method.

5.1. Simulation Method

KONE BTSTM, or BTS in short, is a discrete event simulator, which models building entities such as elevator groups, escalators and stairs. BTS generates passengers for the specified simulation time according to *traffic pattern* and *arrival rate*. Passengers find their way inside the building from their departure floors until arriving on their destination floors. While on route, they use elevators as in real buildings, i.e., press call buttons, wait for service, enter the serving elevator, and travel inside it to their destination floors. BTS records a log of simulation events, which are afterwards processed to obtain performance measures such as passenger waiting times.

A traffic pattern is defined by *traffic components*: incoming, outgoing and interfloor passengers (Siikonen 1993). *Incoming* passengers depart from the *entrance floor* of the building and are destined to an *upper floor* of the building. *Outgoing* passengers are just the opposite to the incoming passengers. *Interfloor* passengers travel only between the upper floors. Typical traffic patterns in office buildings are, e.g., morning up-peak and lunch time mixed traffic.

In *serial* simulations, the arrival rate increases gradually in steps (Siikonen 1993). In each step, a new simulation is started and the simulator generates passengers for 30 minutes with a constant arrival rate. For each step of the serial simulation, the *mean* of a performance measure is calculated by removing a five-minute warm-up period from the observations. Hakonen and Siikonen (2009) found that when mean is defined in this way from a 30-minute simulation, it predicts accurately the steady-state mean.

The traffic patterns used in the simulations are defined in terms of traffic components in Table 2 below. Here the first and the last arrival rates of the serial simulations are given, too. The arrival rate is increased in steps of 1 % of the building population in five minutes. We replicate each serial simulation three times with different random number sequences for each of the 20 simulation cases defined in Appendix B. The last simulated arrival rate varies for the traffic patterns since the

transportation capacity in down-peak and mixed traffic is greater than in up-peak (Barney 2003, Siikonen 1993). In order to reach saturation in all traffic patterns, we simulate arrival rates beyond the theoretical *up-peak handling capacities* of the defined elevator groups, see Appendix A.

Table 2 Traffic definitions of the serial simulations.

Traffic pattern	Traffic components (%)			Arrival rates (% of pop / 5 min)	
	Incoming	Outgoing	Interfloor	First	Last
Up-peak	100	0	0	1	14
Down-peak	0	100	0	1	22
Mixed	40	40	20	1	17

BTS contains algorithms, such as GA and ESP, which also take the transportation requests and state information as input from the simulation. GA uses simple five-minute statistics measured in the simulation as the expected arrival rates λ for each pickup request, recall equation (3.8). The algorithm determines the optimal assignments, which are then used to dispatch elevators to serve transportation requests. The dispatch decisions affect elevator routes in the simulation as well as the realized performance measures. This allows us to compare algorithms on a statistical level.

5.2. Computational Performance of GA

In this section, we study the computational performance of GA in down-peak and mixed traffic situations. Up-peak traffic is not interesting from the computational point of view since an EDP instance consists of at most one registered pickup request, namely, the one upwards at the entrance floor. First, we discuss the convergence of GA when solving single EDP instances. We then analyze statistically the computation times for numerous EDP instances. These results show that GA solves EDP very fast in an on-line fashion.

Convergence of GA. The convergence of GA is shown in Figure 2 below for EDP instances in both down-peak and mixed traffic. In these EDP instances, $E = 8$ elevators are assigned for $|\mathcal{N}^+| = 20$ pickup requests. Thus, the total number of feasible solution alternatives is equal to 8^{20} , which cannot be solved within reasonable time by explicit enumeration of all alternatives. In these cases, the computation time with GA is only 23 milliseconds for down-peak and 92 milliseconds for mixed traffic.

In Figure 2, the minimum, the maximum, and the mean fitness of each generation is scaled by dividing them by the worst fitness value of all generations. In the case of down-peak traffic, the global minimum 0.36195 is found in less than 10 generations. In mixed traffic, the minimum fitness in the 30th generation is 0.35185, which is already close to the optimum 0.34831 but small

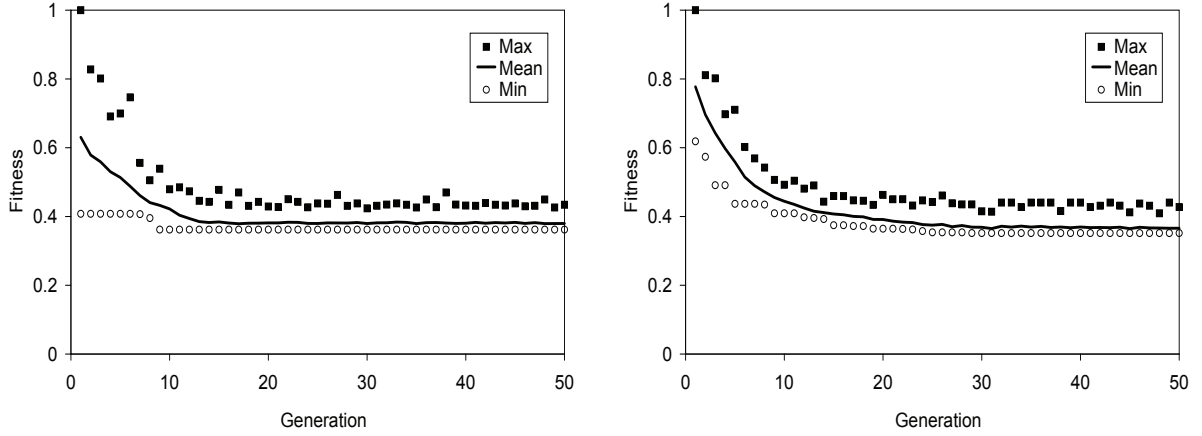


Figure 2 Convergence of GA during the first 50 generations for EDP instances with 20 pickup requests. Left: down-peak traffic; Right: mixed traffic.

improvements still take place in later generations. This, along with longer computation time, shows that the EDP instance in mixed traffic is more difficult to solve than the instance in down-peak traffic.

Computation time of GA. In all of the down-peak and mixed traffic simulations, the total number of solved EDP instances is about $8 \cdot 10^6$. Computation times of these EDP instances are aggregated according to the complexity of the particular instance, i.e., by the number of elevators, E , and pickup requests, $|\mathcal{N}^+|$. The mean and maximum computation times are shown in Figure 3 below for the mixed traffic simulations. Note that the figure below includes only the simulation cases with $E = 4$ and $E = 8$ elevators as examples. The mean computation time stays below 50 milliseconds except for the most demanding problem instances with more than 25 pickup requests. The maximum computation time is less than 100 milliseconds in all but one case.

The mean computation time increases steadily with respect to the number of pickup requests but shows some inconsistency in the most complex cases. The maximum computation times increase more randomly than the means but still about linearly. The number of elevators in the group significantly affects the computation time as can be expected. On average, it takes about 40-50 % longer to solve an instance with eight elevators than with four elevators.

Similar graphs as shown for the mixed traffic can also be obtained for down-peak, but are not shown here to avoid unnecessary repetition. In the case of down-peak, the mean computation time stays below 20 milliseconds and the maximum below 80 milliseconds. This result is in accordance with the previous result that the convergence of GA is slower in mixed than in down-peak traffic instance. However, the computation times are still clearly within the requirements of on-line optimization, and even for the most complex EDP instances.

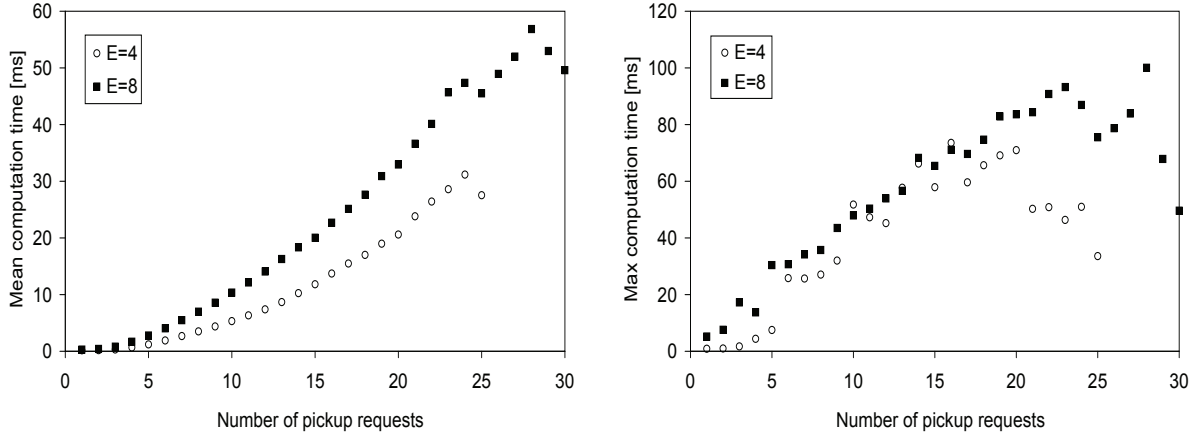


Figure 3 Computation time with respect to the number of pickup requests in mixed traffic. Left: mean; Right: maximum.

5.3. Elevator Group Performance

Now we study both quantity and quality of service of an elevator group by using regression equations. Let κ index the simulation cases shown in Appendix B and let $\mu = 1, \dots, M$ index a serial simulation with arrival rates $\alpha_1, \dots, \alpha_M$, where M denotes the number of simulations. Table 2 in Section 5.1 gives α_1 and α_M as a percentage of building population in five minutes. Let $\bar{Y}_{\kappa,\mu}$ denote the mean of performance measure $Y_{\kappa,\mu}$ of simulation case κ in simulation step μ .

We normalize arrival rates α_μ so that the means of performance measures, $\bar{Y}_{\kappa,\mu}$, are comparable. The normalized arrival rate $\beta_{\kappa,\mu}$, as a percentage of *up-peak handling capacity*, HC_κ , is

$$\beta_{\kappa,\mu} = \alpha_\mu U_\kappa / HC_\kappa, \quad (5.1)$$

where U_κ is the total building population of simulation case κ . By convention, handling capacity expresses the maximum number of passengers an elevator group is capable of transporting in five minutes, see Appendix A. In the following, $\beta_{\kappa,\mu}$ is the independent variable and $\bar{Y}_{\kappa,\mu}$ is the dependent variable of the regression analysis. Note that we have three independent observations of $\bar{Y}_{\kappa,\mu}$ for each $\beta_{\kappa,\mu}$ since we replicate each serial simulation three times.

Next, we show that GA outperforms ESP with respect to both quantity and quality of service. The regression equations, which we derive, play a key role in the comparison.

Quantity of service. Traditional analytical formulas for up-peak handling capacity are derived using probability theory and assuming *utilization factor* $\rho = 0.8$, see Appendix A. We define *transportation capacity* for any traffic situation as the arrival rate $\beta_{\kappa,\mu}$ of a simulation, where the mean utilization factor, $\bar{\rho}_{\kappa,\mu}$, equals 0.8. Now, we derive a regression equation for the utilization factor as

a function of arrival rate to predict this point accurately. In addition, we show that our simulation method is in accordance with the theory.

We first generalize *up-peak round trip*, where an elevator loads P passengers at the main entrance floor, transports them to their destinations and returns to the entrance. *Generalized round trip* consists of the starts of the elevator upwards and downwards until it starts upwards again after reversing its direction of travel twice (Hakonen and Siikonen 2009). Our definition does not depend on the location where the direction of travel changes. In addition, it reduces to the traditional definition in the up-peak situation.

Let $\mathcal{S}_{\kappa,\mu,\nu}$ be the set of elevator starts during round trip ν , indexed by σ . Let $P_{\kappa,\mu,\nu,\sigma}$ denote the number of passengers inside the elevator for start σ in round trip ν . As in Hakonen and Siikonen (2009), we define the utilization factor for round trip ν as the maximum load relative to the rated capacity of an elevator, C_κ ,

$$\rho_{\kappa,\mu,\nu} = \max_{\sigma \in \mathcal{S}_{\kappa,\mu,\nu}} \{P_{\kappa,\mu,\nu,\sigma}\} / C_\kappa. \quad (5.2)$$

An observation of the dependent variable is the mean of $\rho_{\kappa,\mu,\nu}$ over ν round trips, i.e., $\bar{Y}_{\kappa,\mu} = \bar{\rho}_{\kappa,\mu}$.

Figure 4 below shows $\bar{\rho}_{\kappa,\mu}$ of the up-peak and down-peak simulations with GA for cases with $E_\kappa = 4$ and $E_\kappa = 8$ elevators as examples. The graphs suggest logistic growth of $\bar{\rho}_{\kappa,\mu}$ as a function of $\beta_{\kappa,\mu}$. Clearly, also the number of elevators in the group, E_κ , has an effect on $\bar{\rho}_{\kappa,\mu}$. Thus, we estimate regression coefficients for a logistic growth function

$$\rho(\beta, E) = K / (1 + b \exp(-r(\beta, E))), \quad (5.3)$$

where K is the limiting growth size, b is a constant and $r(\beta, E)$ is the growth rate function (Draper and Smith 1998). Note that we use an experimentally found function $r(\beta, E)$ for the growth rate instead of the usual linear relationship. We carry out the regression analysis on the simulation output of all traffic patterns for both GA and ESP, the details of which are shown in Appendix C.

Now we are finally ready to compare GA and ESP with respect to the quantity of service in the up-peak, down-peak and mixed traffic situations. In order to make the comparison, we solve equation (5.3) for β and substitute $\rho = 0.8$. The resulting β is now interpreted as transportation capacity. The comparison is shown in Table 3 below for elevator groups with $E = 4, \dots, 8$ elevators.

In up-peak traffic, the transportation capacity is slightly below the theoretical up-peak handling capacity, i.e., $\beta < 1$, for $E = 5 \dots 8$ elevators. Although the difference is negligible in practice, experience and theory do not expect this. However, we conclude that the transportation capacity in up-peak is consistent with theoretical handling capacity. Note that we do not repeat up-peak

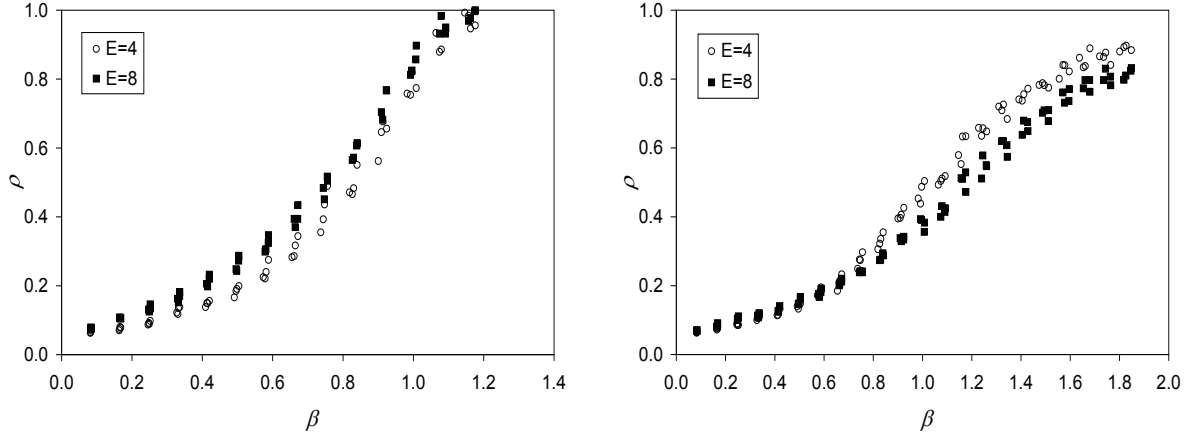


Figure 4 Mean utilization factor $\bar{\rho}_{\kappa,\mu}$ for GA with $E = 4$ and $E = 8$ elevators. Left: up-peak traffic; Right: down-peak traffic.

Table 3 Arrival rate β for $\rho = 0.8$.

	GA			ESP	
	Up-peak	Down-peak	Mixed traffic	Down-peak	Mixed traffic
$E = 4$	1.007	1.547	1.270	1.369	1.230
$E = 5$	0.996	1.591	1.268	1.393	1.236
$E = 6$	0.984	1.635	1.267	1.419	1.242
$E = 7$	0.971	1.681	1.265	1.446	1.248
$E = 8$	0.959	1.728	1.263	1.475	1.254

simulations for ESP since we assume that these algorithms have equal transportation capacity in up-peak.

On the other hand, in down-peak traffic the efficiency of GA is clearly seen. GA increases transportation capacity by about 15 % compared to ESP. Thus, the algorithm has the greatest effect on the transportation capacity in down-peak. The reason behind this fact is that down-peak leaves the greatest freedom for the algorithm to optimize the elevator routes. Finally, GA is also a bit more efficient than ESP in mixed traffic but the difference is negligible in practice.

Quality of service. We define *waiting time* of passenger ν , $W_{\kappa,\mu,\nu}$, as the time from either registering a pickup request or joining the lobby queue until serving elevator starts to open its doors (Barney et al. 2005). In order to compare the quality of service between elevator groups, the waiting time is normalized by the theoretical *up-peak interval*, INT_{κ} , see Table 5 in Appendix B. Thus, our performance measure for quality of service is

$$\omega_{\kappa,\mu,\nu} = W_{\kappa,\mu,\nu} / INT_{\kappa}, \quad (5.4)$$

An observation of the dependent variable is the mean over all passengers in simulation step μ of

simulation case κ , i.e., $\bar{Y}_{\kappa,\mu} = \bar{\omega}_{\kappa,\mu}$. These performance measures of GA and ESP in both down-peak and mixed traffic are shown in Figure 5 below. Only simulations with $E = 8$ elevators are shown in order to make the figure legible. The figure suggests exponential growth,

$$\omega(\beta, E) = \exp(r(\beta, E)). \quad (5.5)$$

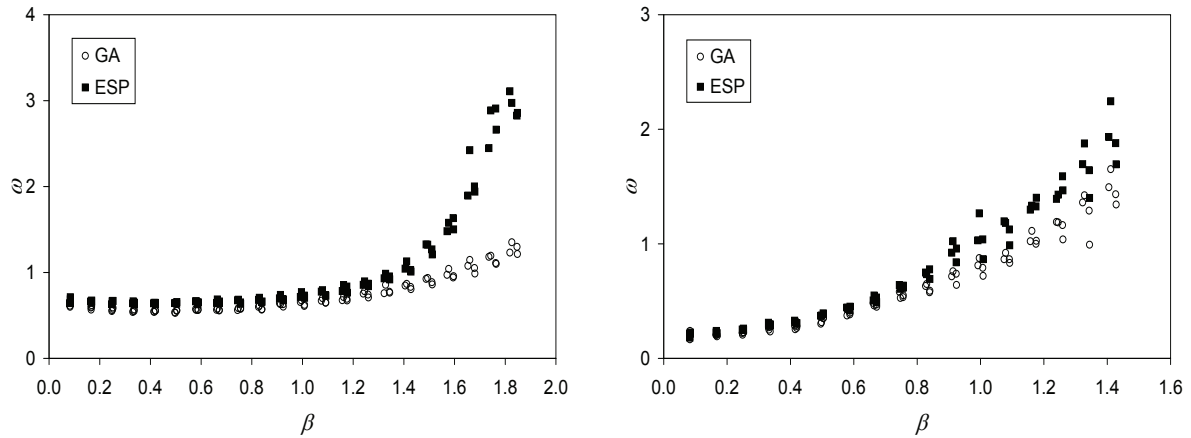


Figure 5 Quality of service $\bar{\omega}_{\kappa,\mu}$ for GA and ESP. Left: down-peak; Right: mixed traffic.

Again, we carry out the regression analysis, the details of which are presented in Appendix C. Table 4 below compares the predicted quality of service ω of GA and ESP after substituting arrival rate $\beta = 1$ in equation (5.5). In down-peak, the quality of service when using GA is 10-16 % better than when using ESP. On the other hand, in mixed traffic the improvement is 13-20 %. Clearly, the impact of the use of GA to the elevator group performance is dramatic.

Table 4 Quality of service ω for $\beta = 1$.

	GA			ESP	
	Up-peak	Down-peak	Mixed traffic	Down-peak	Mixed traffic
$E = 4$	0.570	0.594	0.837	0.709	0.961
$E = 5$	0.506	0.604	0.825	0.708	0.965
$E = 6$	0.450	0.614	0.813	0.706	0.969
$E = 7$	0.399	0.624	0.801	0.705	0.973
$E = 8$	0.354	0.634	0.789	0.703	0.977

Siikonen (1993) concluded that in down-peak average waiting times start to increase rapidly at arrival rates $\beta = 1.6, \dots, 1.8$ when using ESP-algorithm. This, on the other hand, could be used as a criteria for defining the quantity of service. Our simulations are consistent with her conclusion,

see Figure 5. However, our results show that arrival rate $\beta \approx 1.4$ is the limit for ESP, where the mean utilization factor exceeds 0.8, see Table 3. Thus, the difference of these conclusions comes from the definitions.

Example of using regression equations. Next, we demonstrate the use of the regression equations. Here we consider only GA. Let an elevator group be given with $E = 8$ elevators, $Z = 24$ upper floors, up-peak handling capacity $HC = 6.5$ % of population in five minutes and up-peak interval $INT = 25.6$ seconds. These parameters are sufficient for evaluating elevator group performance when planning a building. Traditionally, simulation is used to evaluate performance also in down-peak and mixed traffic. The derived regression equations (5.3) and (5.5) provide an alternative method for the evaluation. These methods are compared in Figure 6 below, where the continuous curves predict the performance using the regression equations and the dots show the simulated performance measures.

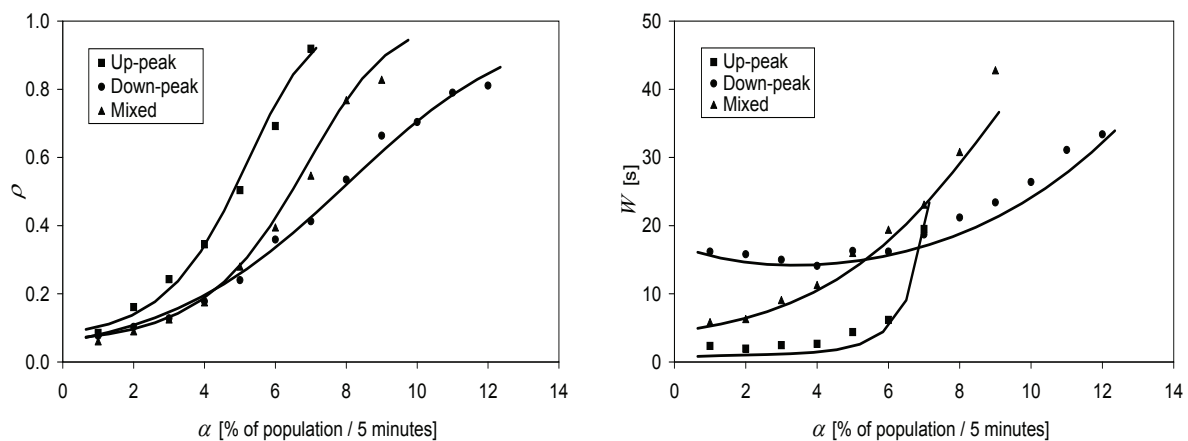


Figure 6 Predicted and simulated elevator group performance in up-peak, down-peak and mixed traffic. Left: mean utilization factor; Right: mean waiting time.

According to the figure, the simulated values of the utilization factor and waiting time closely follow the predicted curves. The figure also clearly shows how the elevator group performance depends on the traffic pattern.

6. Discussion

In this paper, we give a thorough treatment of the Elevator Dispatching Problem, EDP. For the first time, the full mathematical formulation is given, accompanied with an efficient on-line algorithm, which is used in KONE AltaTM high-rise elevators. We formulate EDP as a stochastic bilevel programming problem. We solve it to optimality by iterating upper level assignments, for which

a heuristic algorithm defines the optimal route for each lower level problem. Using simulations, we measure computation times of the algorithm. Furthermore, we derive regression equations to predict both quantity and quality of service in up-peak, down-peak and mixed traffic.

The use of a genetic algorithm, GA, in a real-time control has been verified by numerous installations of KONE AltaTM elevators. Extensive simulations presented in this paper show that GA solves even large problem instances fast, satisfying the requirements of on-line implementation. It should be noted here that industrial PCs, which run the algorithm in real installations, are less powerful than the hardware used for the measurements in this paper. However, the problem instances studied here are still solvable within 500 milliseconds with usual hardware.

The regression equation for utilization factor allows us to define quantity of service in a general traffic situation from simulation output. Our result from up-peak simulations is consistent with up-peak handling capacity, which is predicted analytically by probability theory. When using GA, transportation capacity in down-peak is 54-73 % and in mixed traffic about 26 % greater than in up-peak, depending on the number of elevators in the group. For the earlier rule-based Enhanced Spacing Principle, ESP, (Siikonen 1997), these values are 37-48 % in down-peak and about 24 % in mixed traffic. Barney (2003) shows similar results but does not use simulation. Rather, she estimates down-peak and mixed traffic handling capacity in a similar manner as up-peak handling capacity using over-simplifying assumptions. Her approach does not take into account the effect of the algorithm.

GA also outperforms earlier algorithms with respect to the quality of service, which is defined as the mean waiting time of simulation normalized by theoretical up-peak interval. For down-peak traffic at an arrival rate corresponding to up-peak handling capacity, the quality of service with GA is about 0.6, while it is 0.7 for ESP and 0.85 for the algorithm of Barney (2003). For mixed traffic, these values are about 0.81 with GA and about 0.97 with ESP. Barney (2003) does not report results for mixed traffic in a comparable manner. Thus, average waiting times with GA are about 15 % shorter than with ESP and about 30 % shorter than with the algorithm used in Barney (2003).

Since our mathematical formulation of EDP takes the essential uncertainties into account for the first time, it also stands for a starting basis for research of alternative methods and formulations. A mixed integer programming formulation would be the first step to solve EDP instances with exact algorithms, which then would act as a benchmark for the current algorithm. Also other heuristic algorithms, which have been used successfully in conjunction with vehicle routing problems such as tabu search (Gendreau et al. 1999), could be used instead of the genetic algorithm. If more choices

were needed to fine-tune the operation of the elevators, an alternative genetic algorithm could be implemented to handle multiple objectives (Deb et al. 2002).

We considered only stochastic demand in our EDP formulation. In addition, stochastic customers could be added to the formulation. Then, a method to forecast new arrivals in the near future as well as a state-of-the-art stochastic optimization is needed. Such a forecasting method has not been studied earlier and deserves its own research. Full stochastic optimization is probably out of the question for an on-line implementation but a robust counterpart, e.g., Ben-Tal and Nemirovski (2002) or Bertsimas and Sim (2003), could be implemented within the computational requirements.

Our approach opens up the possibility to derive analytic optimality conditions for the nearest neighbor heuristic in the context of one-dimensional traveling salesman problem, which we shall study in a subsequent paper. It remains to be seen whether this result can be generalized for two-dimensional TSPs. We hope that our work encourages researchers to use optimization-based methods on hard real world problems requiring on-line implementation. Even if formulation, algorithm and on-line implementation need to be considered all together, such challenges are solvable as our example shows.

Appendix A: Up-Peak Performance

A wealth of theoretical results has been found for morning up-peak traffic, which is used as a fundamental benchmark in elevator planning. Here we summarize the most important results. *Up-peak analysis* defines the maximum number of passengers an elevator group can transport in five minutes. Traditionally, average elevator load 80 % of the *rated capacity*, C , is assumed in this analysis. Using queuing theory, Alexandris (1977) showed that both queue length and passenger waiting times start to increase rapidly if the average *utilization factor*, or *load factor*, exceeds 0.8. This result has been verified by numerous simulation studies.

Up-peak analysis is based on the expected *round trip time*, RTT . In up-peak, an elevator *round trip* starts when it opens its doors at the main entrance floor. Then, $P = 0.8C$ passengers enter the elevator and are transported to their destinations. The round trip ends when the elevator returns to the main entrance floor and opens its doors again. Alexandris (1977) derived the expected number of stops, S , and the expected reversal floor, H , for up-peak round trip using probability theory. Finally, the expected round trip time as defined in Alexandris (1977), is

$$RTT = 2Ht_v + (S + 1)t_s + 2Pt_p, \quad (\text{A.1})$$

where t_v denotes the run time of one floor with *nominal speed* v , t_s denotes the stop time, and t_p denotes the passenger transfer time, i.e., the time it takes to either enter or exit the elevator. Note that here t_s includes the time to decelerate from the nominal speed to rest, door opening and closing times as well as other delays, and the time to accelerate.

The maximum number of passengers the elevator group can transport in five minutes, i.e., *quantity of service*, is defined by up-peak *handling capacity*,

$$HC = 0.8 \cdot 300 \cdot E \cdot C / RTT, \quad (\text{A.2})$$

where E denotes the number of elevators in the group. Note also the use of utilization factor 0.8 in this definition. Usually, handling capacity is normalized by the total served population, U , and expressed as a percentage of population in five minutes.

Quality of service is defined by up-peak *interval*, INT , which is the expected time between successive elevator departures from the entrance floor,

$$INT = RTT / E. \quad (\text{A.3})$$

Interval correlates with passenger waiting times.

Elevator planning is based on handling capacity and interval. In office buildings with multiple tenants, handling capacity should be at least about 12% of the population in five minutes, and interval should stay below 40 seconds (Siikonen 2001). Similar, but not identical, criteria are given in, e.g., Barney (2003).

Appendix B: Simulation Cases

We define 20 different elevator groups with the parameters shown in Table 5 below. Each of the defined elevator groups fulfils the above-mentioned up-peak criteria. An elevator group serves zone $\mathcal{Z} = \{0, \dots, Z\}$, where 0 denotes the only entrance floor and the Z upper floors are immediately above it. Note that the table below shows the number of upper floors, Z , but the total number of floors in the zone is $Z + 1$.

Other shared parameters among all the elevator groups are floor-to-floor distance 3.3 m, rated capacity $C = 17$ persons, acceleration 1.0 m/s^2 , jerk 1.6 m/s^3 , door opening time $t_o = 1.4 \text{ s}$, door closing time $t_c = 3.1 \text{ s}$, advance door opening time 0.5 s , other delays 1.6 s , and passenger transfer time $t_p = 1.0 \text{ s}$ to either enter or exit the elevator. Elevator capacity Q of the certainty equivalent control problem (3.11) refers to a common elevator system parameter called bypass load. We use bypass load $Q = 0.8C$.

In evaluating up-peak round trip time, we use the equation derived in Roschier and Kaakinen (1979) instead of (A.1). The former is more accurate than the latter, especially in cases where floor heights differ or some of the upper floors have no occupants.

Appendix C: Results of Regression Analysis

The regression equation for utilization factor ρ is

$$\rho(\beta, E) = 1 / (1 + b \exp(-r(\beta, E))), \quad (\text{C.1})$$

where $b = 1/\bar{\rho}_{\kappa, \bar{\mu}} - 1$ is defined as the starting growth value of the utilization factor, $\bar{\mu} = \arg \min_{\mu} \beta_{\kappa, \mu}$, for all κ , and

$$r(\beta, E) = r_0 + r_1\beta + r_2\beta^2 + E(r_3 + r_4\beta + r_5\beta^2). \quad (\text{C.2})$$

The limiting growth size K above is 1 since we allow full elevator loads in the simulation. Thus, we estimate coefficients r_0, \dots, r_5 by using regression analysis, the results of which are shown in Table 6 below. The F-test

Table 5 Elevator group parameters of each simulation case κ .

κ	E (elevators)	Z (floors)	U (persons)	v (m/s)	RTT (s)	INT (s)	HC (persons / 5 min)
1	4	16	880	2.5	155.8	38.9	105
2	4	18	846	3.0	160.2	40.0	102
3	4	20	820	3.5	164.5	41.1	99
4	4	22	792	4.0	168.7	42.2	97
5	5	16	1104	2.5	155.8	31.2	131
6	5	18	1062	3.0	160.2	32.0	127
7	5	20	1040	3.5	164.5	32.9	124
8	5	22	1012	4.0	168.7	33.7	121
9	6	16	1296	2.5	155.8	26.0	157
10	6	18	1260	3.0	160.2	26.7	153
11	6	20	1240	3.5	164.5	27.4	149
12	6	22	1210	4.0	168.7	28.1	145
13	7	16	1520	2.5	155.8	22.3	183
14	7	18	1476	3.0	160.2	22.9	178
15	7	20	1460	3.5	164.5	23.5	174
16	7	22	1408	4.0	168.7	24.1	169
17	8	16	1760	2.5	155.8	19.5	210
18	8	18	1710	3.0	160.2	20.0	204
19	8	20	1640	3.5	164.5	20.6	198
20	8	22	1606	4.0	168.7	21.1	193

Table 6 Results of regression analysis for utilization factor ρ .

Parameter	GA			ESP	
	Up-peak	Down-peak	Mixed traffic	Down-peak	Mixed traffic
Observations	762	1320	1020	1320	1020
R^2	0.982617	0.99394	0.991704	0.988507	0.991613
Standard Error	0.188646	0.110387	0.139423	0.176616	0.144643
Significance F	0	0	0	0	0
r_0	-0.35819	-0.069554	-0.38547	-0.89253	-0.1824
r_1	0.693753	3.786807	1.811809	3.869242	1.230464
r_2	3.339062	-0.24014	1.292703	N/A	1.92872
r_3	0.087701	0.080118	0.050435	0.125223	0.036746
r_4	N/A	-0.21094	-0.14068	-0.00205	-0.05669
r_5	N/A	0.052158	0.084766	0.038942	N/A
b	14.79779	15.12903	14.10574	14.72327	15.36661

and the R^2 statistics show that the regression is statistically significant. Note that some of the coefficients are not statistically significant in all cases, which is denoted by 'N/A'.

The regression equation for quality of service ω is

$$\omega(\beta, E) = \exp(r(\beta, E)), \quad (\text{C.3})$$

where the growth rate function is

$$r(\beta, E) = r_0 + r_1\beta + r_2\beta^2 + r_3\beta^3 + E(r_4 + r_5\beta + r_6\beta^2). \quad (\text{C.4})$$

We estimate the coefficients r_0, \dots, r_6 using regression analysis, which is again statistically significant. The results of the analysis are shown in Table 7 below.

Table 7 Results of regression analysis for quality of service ω .

Parameter	GA			ESP	
	Up-peak	Down-peak	Mixed traffic	Down-peak	Mixed traffic
Observations	762	1320	1020	1320	1020
R^2	0.917081	0.980382	0.976904	0.968484	0.975275
Standard Error	0.322359	0.051468	0.108274	0.11016	0.119925
Significance F	0	0	0	0	0
r_0	-7.02308	-1.828	-2.41896	-2.02319	-2.4617
r_1	18.17895	0.968082	2.324384	2.12489	2.769419
r_2	-16.9144	0.418017	0.405572	-0.94577	N/A
r_3	5.673129	-0.14543	-0.43022	0.508059	-0.36545
r_4	0.421039	0.179476	0.083845	0.205303	0.09838
r_5	-1.94301	-0.2212	-0.17484	-0.26659	-0.21278
r_6	1.403021	0.058336	0.076298	0.059319	0.118765

References

- Alexandris, N.A. 1977. Statistical models in lift systems. Ph.D. thesis, The Victoria University of Manchester, Institute of Science and Technology.
- Barney, G., R. Peters, B. Powell, M-L. Siikonen. 2005. Towards agreed traffic design definitions. *Elevator World* **53**(2) 108.
- Barney, G.C. 2003. *Elevator Traffic Handbook*. Spon Press, London.
- Ben-Tal, A., A. Nemirovski. 2002. Robust optimization - methodology and applications. *Math. Programming, Ser.B* **92**(3) 453–480.
- Bertsekas, D.P. 2005. *Dynamic Programming and Optimal Control*, vol. I. 3rd ed. Athena Scientific, Belmont, MA.

- Bertsimas, D.J., P. Jaillet, A.R. Odoni. 1990. A priori optimization. *Oper. Res.* **38**(6) 1019–1033.
- Bertsimas, D.J., M. Sim. 2003. Robust discrete optimization and network flows. *Math. Programming* **98**(1) 49–71.
- Bertsimas, D.J., G. van Ryzin. 1991. A stochastic and dynamic vehicle routing problem in the euclidean plane. *Oper. Res.* **39**(4) 601–615.
- Closs, G.D. 1970. The computer control of passenger traffic in large lift systems. Ph.D. thesis, The Victoria University of Manchester, Institute of Science and Technology.
- Deb, K., A. Pratap, S. Agarwal, T. Meyarivan. 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation* **6**(2) 182–197.
- Draper, N.R., H. Smith. 1998. *Applied regression analysis*. 3rd ed. John Wiley & Sons, Inc, New York, NY.
- Gendreau, M., F. Guertin, J-Y. Potvin, É. Taillard. 1999. Parallel tabu search for real-time vehicle routing and dispatching. *Transportation Sci.* **33**(4) 381–390.
- Gendreau, M., G. Laporte, R. Séguin. 1995. An exact algorithm for the vehicle routing problem with stochastic customers and demands. *Transportation Sci.* **29**(2) 143–155.
- Gendreau, M., G. Laporte, R. Séguin. 1996. Stochastic vehicle routing. *Eur. J. Oper. Res.* **88**(1) 3–12.
- Goldberg, D.E. 1989. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, Boston.
- Hakonen, H., M-L. Siikonen. 2009. Elevator traffic simulation procedure. *Elevator World* **57**(9) 180–190.
- Hirasawa, K., S. Kuzunuki, T. Iwasaki, T. Kaneko. 1978. Optimal hall call assignment method of elevator group supervisory control system. *Joint Automatic Control Conference*. USA, 305–313.
- Ichoua, S., M. Gendreau, J-Y. Potvin. 2000. Diversion issues in real-time vehicle dispatching. *Transportation Sci.* **34**(4) 426–438.
- Ichoua, S., M. Gendreau, J-Y. Potvin. 2003. Vehicle dispatching with time-dependent travel times. *Eur. J. Oper. Res.* **144**(2) 379–396.
- Jaillet, P. 1988. A priori solution of a traveling salesman problem in which a random subset of the customers are visited. *Oper. Res.* **36**(6) 929–936.
- Kavounas, G.T. 1993. Designing elevator systems also for the average waiting time. *Elevator World* **41**(1) 100–103.
- Laporte, G., F. Louveaux, H. Mercure. 1992. The vehicle routing problem with stochastic travel times. *Transportation Sci.* **26**(3) 161–170.
- Laporte, G., F.V. Louveaux. 1993. The integer L-shaped method for stochastic integer programs with complete recourse. *Operations Research Letters* **13**(3) 133–142.
- Laporte, G., F.V. Louveaux, H. Mercure. 1994. A priori optimization of the probabilistic traveling salesman problem. *Oper. Res.* **42**(3) 543–549.

- Larson, R.C., A.R. Odoni. 2007. *Urban Operations Research*. Dynamic Ideas, Belmont, MA.
- Levy, D., M. Yadin, A. Alexandrovitz. 1977. Optimal control of elevators. *Int. J. Systems Sci.* **8**(3) 301–320.
- Markon, S., H. Kita, H. Kise, T. Bartz-Beielstein. 2006. *Control of Traffic Systems in Buildings*. Springer.
- Markon, S., H. Kita, Y. Nishikawa. 1994. Adaptive optimal elevator group control by use of neural networks. *Transactions of the Institute of Systems, Control and Information Engineers* **7**(12) 487–497.
- Miller, E.D., H.S. Mahmassani, A. Ziliaskopoulos. 1994. Path search techniques for transportation networks with time-dependent, stochastic arc costs. *IEEE International Conference on Systems, Man, and Cybernetics. Humans, Information and Technology*. **2** 1716–1721.
- Pepyne, D.L., C.G. Cassandras. 1997. Optimal dispatching control for elevator systems during uppeak traffic. *IEEE Transactions on Control Systems Technology* **5**(6) 629–643.
- Picard, J-C., M. Queyranne. 1978. The time-dependent traveling salesman problem and its application to the tardiness problem in one-machine scheduling. *Oper. Res.* **26**(1) 86–110.
- Psaraftis, H.N. 1988. Dynamic vehicle routing problems. B.L. Golden, A.A. Assad, eds., *Vehicle Routing: Methods and Studies*. North Holland, 223–248.
- Psaraftis, H.N., J.N. Tsitsiklis. 1993. Dynamic shortest paths in acyclic networks with markovian arc costs. *Oper. Res.* **41**(1) 91–101.
- Roschier, N-R., M.J. Kaakinen. 1979. New formulae for elevator round trip calculation. *Supplement to Elevator World for ACIST Members* 189–197.
- Shimizu, K., Y. Ishizuka, J.F. Bard. 1997. *Nondifferentiable and Two-Level Mathematical Programming*. Kluwer Academic Publishers, Dordrecht.
- Siikonen, M-L. 1993. Elevator traffic simulation. *Simulation* **61**(4) 257–267.
- Siikonen, M-L. 1997. Planning and control models for elevators in high-rise buildings. Ph.D. thesis, Helsinki University of Technology, Systems Analysis Laboratory.
- Siikonen, M-L. 2001. On traffic planning methodology. *Lift Report* **27**(3) 24–29.
- Siikonen, M-L., T. Susi, H. Hakonen. 2001. Passenger traffic flow simulation in tall buildings. *Elevator World* **49**(8) 117–123.
- Sorsa, J., M-L. Siikonen, H. Ehtamo. 2003. Optimal control of double-deck elevator group using genetic algorithm. *Intl. Trans. in Oper. Res.* **10**(2) 103–114.
- Swihart, M.R., J.D. Papastavrou. 1999. A stochastic and dynamic model for the single-vehicle pick-up and delivery problem. *Eur. J. Oper. Res.* **114**(3) 447–464.
- Sörensen, K. 2006. Route stability in vehicle routing decisions: a bi-objective approach using metaheuristics. *Central European Journal of Operations Research* **14**(2) 193–207.
- Tagmouti, M., M. Gendreau, J-Y. Potvin. 2007. Arc routing problems with time-dependent service costs. *Eur. J. Oper. Res.* **181**(1) 30–39.

- Tanaka, S., Y. Uruguchi, M. Araki. 2005. Dynamic optimization of the operation of single-car elevator systems with destination hall call registration: Part I. Formulation and simulations. *Eur. J. Oper. Res.* **167**(2) 550–573.
- Tyni, T., J. Ylinen. 1999. Improving the performance of genetic algorithms with a gene bank. K. Miettinen et al, ed., *Proceedings of EUROGEN99 - Short Course on Evolutionary Algorithms in Engineering and Computer Science*. A 2/1999, University of Jyväskylä, Department of Mathematical Information Technology, 162–170.
- Tyni, T., J. Ylinen. 2001. Genetic algorithms in elevator car routing problem. L. Spector et al., ed., *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)*. Morgan Kaufman Publishers, San Francisco, CA, USA, 1413–1422.
- Tyni, T., J. Ylinen. 2006. Evolutionary bi-objective optimisation in the elevator car routing problem. *Eur. J. Oper. Res.* **169**(3) 960–977.
- Wesselowski, K.S., C.G. Cassandras. 2006. The elevator dispatching problem: Hybrid system modeling and receding horizon control. C. Cassandras, A. Giua, C. Seatzu, J. Zaytoon, eds., *Proceedings of 2nd IFAC Conference on Analysis and Design of Hybrid Systems*. Elsevier, Alghero, Italy, 136–141.