# Need a Lift? An Elevator Queueing Problem

Problem presented by
Arthur Hsu, Robert LaBarre, and Slaven Stricevic
UTRC

Participants:

Richard Braun, Bogdan Doytchinov, Alistair Fitt
Stephen Pennell, Colin Please, Ravi Srinivasan
Hermann Servatius, Thomas Witelski

Report prepared by R. Braun, received 14 August 2003.

**Abstract**

Various aspects of the behavior and dispatching of elevators (lifts) were studied. Monte Carlo simulation was used to study the statistics of the several models for the peak demand at uppeak times. Analytical models problems were used to prove or disprove whether schemes were optimal. A mostly integer programming problem was formulated but not studied further.

## 1 Introduction

The proposed problem was to study the efficient dispatching of people requesting service from an elevator in a high-rise building. This is a multifaceted problem, which is too large to attempt in its entirety during such a short workshop. One reason is that the flow of people through a building takes on different characteristics depending on such things as: building use (office, apartment, commercial retail, etc.), time of day, existence of "attraction" floors (e.g., parking garages, cafeteria) and other factors. In particular, there are three main fundamental modes of traffic: uppeak, heavy two-way, and down-peak. This report focuses, as do most studies, on uppeak traffic. This mode has the distinct advantages of having the elevators assumed to be leaving from the lobby, where all passengers are assumed to arrive, and the elevators return rapidly to the lobby after delivering passengers.

Current approaches to improving dispatching are centered around discrete event simulations, taking into account things like passenger arrival rates, service requests from floor landings (hall calls) and from inside cars (car calls), motion profiles of the car, including acceleration and deceleration rates, loading and unloading times, number of elevator cars, capacity of the car, etc. The approaches during the meeting simplified the parameters used to determine the round trip time; a constant speed was used to model the describe the motion of the elevator, there may be a delay time in the lobby, and in the vast majority of what follows, the stop time per floor will be zero.

The goal of the workshop for this problem is to attempt to understand dispatching from a more-analytic perspective, and to understand how elevator dispatching fits into the world of other routing problems. Progress was made in classifying some simplified dispatching models in the context of probability and queueing theory. Some suggestions were made for schemes to try to optimize performance given a priori knowledge of the passenger arrivals and destinations. A counterexample to a hypothesized method of elevator reduction (that would maintain optimal mean waiting time performance) was found. A number of simulations were performed and the resulting elevator demand computed; robust results for the mean number of elevators required and the distribution of the elevators required were established.

The report is organized as follows: Section 2 contains analytical probabilistic approaches, including queueing theory. Section 3 contains some direct analytical treatments, including a counterexample to the marginal reduction algorithm and a proposed clustering approach. Section 4 contains a mixed integer programming formulation, and Section 5 contains simulation results using a Monte Carlo algorithm. The Appendix contains an algorithm for the simulations.

## 2 Analytical Probabilistic Approaches

### 2.1 Queueing Theory

#### 2.1.1 Origins

Queueing theory originated about a century ago in the study of telephony. Later, the theory was much expanded and elaborated, and is now a large branch of mathematics. A queueing system consists of one or more servers, and a flow of customers that come to the system, wait for service, and then leave.

A good introduction to the subject, with a lot of information, is [1].

The pioneer investigator in the theory of queuing systems was the Danish mathematician A. K. Erlang, who published works on the subject in 1909.

Later, one should note the works of Molina in 1927. In the early 1930's significant work was done by Felix Pollaczek. In the same time, similar work was carried out in Russia by Kolmogorov and Khintchine, in France by Crommelin, and in Sweden by Palm.

The basic works that started the consideration of heavy traffic limits of queuing systems belong to Iglehart and Whitt, Reiman. These authors mention as their precursors Kingman, Prohorov, Borovkov. The subject of queues in heavy traffic is being studied intensely.

Harrison initiated the study of queues in heavy traffic connected in a network, and because of their practical importance and computational tractability, these ideas have generated a large body of research.

The situation described in Section 5.1.1 is analogous to the following queueing system studied by Erlang. Suppose we have infinitely many phone lines. At random moments in time, different customers make calls, and each call utilizes the first phone line that is free at the moment. Since there are infinitely many available lines, there is no queue. We are interested in the number of busy lines at any moment.

### 2.1.2 What is the right Queueing System?

There are many types of queueing systems. Which one should we choose as our model?

Let's start with a quick look at the classification and the different types of queueing systems that have been studied.

When describing a queueing system, we need to specify the arrival flow, the service time characteristics, the number of servers, and the queue capacity and discipline.

Arrivals occur at random times, and the interarrival times are usually considered independent. When the interarrival times are exponentially distributed, the arrival flow of customers is a Poisson process. It is easier to study because it is Markov.

Service time is also random, and, again, if it is exponential, the process becomes Markov which simplifies the study.

If we denote the number of servers by $s$, and the queue capacity (how many customers can wait in queue) by $c$, we get different types of queueing systems; here are just some examples:

- $M|M|s|c$ Exponential i.i.d. interarrival times, exponential service time, $s$ servers, and a maximum of $c$ customers allowed in queue.
- $M|M|s$ Exponential i.i.d. interarrival times, exponential service time, $s$ servers, no restriction for the number of customers allowed in queue.
- $M|M|\infty$ Exponential i.i.d. interarrival times, exponential service time, infinitely many servers.
- $M|G|s|c$ Exponential i.i.d. interarrival times, general service time, $s$ servers, and a maximum of $c$ customers allowed in queue.
- $M|D|s|c$ Exponential i.i.d. interarrival times, deterministic (fixed) service time, $s$ servers, and a maximum of $c$ customers allowed in queue.
- $GI|M|s$ General independent interarrival times, exponential service time, $s$ servers, no restriction for the number of customers allowed in queue.
- $GI|G|s|c$ General independent interarrival times, general service time, $s$ servers, and a maximum of $c$ customers allowed in queue.
- $GI|G|\infty$ General independent interarrival times, general service time, infinitely many servers.

All these are queues in which customers are served on a "First Come - First Served" basis. There exist models for priority queues, where the service discipline is different. There are other letters and notations for describing these and other, yet more complicated, situations. A good reference is [1], Section 1.3 (pages 8 and 9).

The simplest possible queue is the $M|M|1$ queue. All queues of the type $M|M|s|c$ (or $M|M|s$, or $M|M|\infty$) are described by Markov processes which makes them easy to study. A lot of results are readily available in the textbooks (see, e.g. [1]). The $M|G|1$ queue is harder, but can be reduced to a Markov process, by augmenting the variable space appropriately. The $M|G|s$ queues are harder to study yet, because the Markov property is lost. In the case of $M|G|\infty$ however, there is a simple description in terms of mixtures of Poisson processes (see, e.g. [1], page 302). We exploit these ideas in Section 2.2

Which queueing system should we choose in order to model the situation of section 5.1.1? It is clear that we have infinitely many servers, and hence no queue. Let's assume that the arrival process is Poisson. Then we have a $M|G|\infty$ queueing system. If we also assume exponential "service time", we will reduce the system to $M|M|\infty$, which is extremely simple (see, e.g. [1], Section 2.6, pages 102–104).

How realistic are these assumptions?

Poisson arrivals are a good approximation in many situations. Often times, a compound Poisson is more appropriate, which means that passengers arrive in batches. The interarrival times are still exponential, but each arrival consists of a random number of passengers. As long as this random number is not greater than the elevator capacity, we can treat these "batches" as passengers, and still use our model with a bit of fudging.

Exponential service times are more problematic. It should be noted that, while it makes the analytical exploration easier, the exponential distribution is not very often encountered in reality. In some limited situations it gives an adequate approximation (see, e.g. [4], page 277; in particular, cf. Fig. 11.1(a) and Fig. 11.1(e)). However, in our situation we can only assume a general distribution for the service time.

Observe that when we talk about "service time" in this model, we mean the time it takes for the elevator to pick up (or "duration") the passenger, take him to the desired floor, and come back. If a given passenger is going to a specific floor $i$, the return-trip time of the elevator is a known positive number $D_i$. For a random passenger, the "service time" will be a random variable which takes values $D_1$, $D_2$, $D_3$, ... with some (known) probabilities.

From this discussion, it becomes clear that an $M|G|\infty$ queue will be a good model for our situation, while an $M|M|\infty$ queue would be too simplistic.

In Section 2.2 we will discuss the $M|G|\infty$ model in some detail. But before this, we will also take a quick look at the $M|M|\infty$ model, exactly because it is so simple and well-studied.

Finally, we wish to note that there are yet other queueing models which could be even more appropriate for describing elevator traffic, namely queues with batch arrivals and batch service. We did not look at all at these during the workshop (due to lack of time), but these models have been studied in the mathematical community, and results can be found in the literature (see e.g. [1], Sections 3.1 and 3.2, pages 156 – 170).

### 2.1.3 Poisson Process

Let $T_1, T_2, T_3, \dots$ be i.i.d. exponential random variables with parameter $\lambda$, i.e.

$$\mathbf{P}\{T_i > x\} = e^{-\lambda x}, \qquad x \geq 0.$$

These random variables will represent the interarrival times between the customers. For the exponential distribution, we have

$$\mathbf{E}T_i = \frac{1}{\lambda},$$

so, this means an arrival rate of $\lambda$ customers per unit time on average.

Let's look now at the dynamic of the arrivals. At first, the queue is empty. After a random time $T_1$, the first customer arrives. The second one arrives at a time which is $T_2$ time units later then the first arrival, etc. Thus, the times of arrivals are $T_1$, $T_1 + T_2$, $T_1 + T_2 + T_3$, etc.

Let's denote by $N_t$ the number of customers arrived by time $t$, i.e.

$$N_t := \max\{k : T_1 + T_2 + \cdots + T_k \leq t\}.$$

With probability 1, the above quantity is finite.

Thus defined, $N$ is a random process. It is called a *Poisson process* with rate $\lambda$.

The Poisson process is well known and well studied and has some nice properties that make it easy to use:

1. $N_0 = 0$ with probability 1.

2. It is a "pure jump" process; it increases by 1 at each of the arrival times, $T_1$, $T_1+T_2$, $T_1 + T_2 + T_3$, ..., and is constant at all other times.

3. It has independent increments. Given an arbitrary positive integer $m$ and arbitrary (deterministic) non-negative times $t_1 < t_2 < t_3 < \cdots < t_m$, the random variables $N_{t_1}$, $N_{t_2} - N_{t_1}$, $N_{t_3} - N_{t_2}$, ..., $N_{t_m} - N_{t_{m-1}}$ are independent.

4. For each $t > 0$, the random variable $N_t$ has a Poisson distribution with parameter $\lambda t$. For $0 < s < t$, the random variable $N_t - N_s$ has a Poisson distribution with parameter $\lambda(t - s)$.

We now turn to the study of the Poisson distribution.

### 2.1.4 Poisson Distribution

The distribution of a Poisson random variable $X$ with parameter $\lambda$ is given by the formula

$$\mathbf{P}\{X = k\} = \frac{\lambda^k}{k!} e^{-\lambda}.$$

We have

$$\mathbf{E}X = \text{var}X = \lambda.$$

If $X_1$ and $X_2$ are independent Poisson random variables, with parameters $\lambda_1$ and $\lambda_2$ respectively, then $X_1 + X_2$ is a Poisson random variable with parameter $\lambda_1 + \lambda_2$.

For relatively large $\lambda$, the distribution of

$$\frac{X - \lambda}{\sqrt{\lambda}}$$

is approximated well by a standard normal distribution. This can be interpreted as an instance of the Central Limit Theorem (since the Poisson Distribution is infinitely divisible), but it is easier to prove directly, using generating functions.

It is interesting to note that there exists a "functional" version of this. If $N_t$ is a Poisson Process with rate $\lambda$, then, for large values of $\lambda$,

$$\frac{N_t - \lambda t}{\sqrt{\lambda}} \tag{1}$$

is well approximated in distribution by a standard Wiener process (Brownian motion). This follows from Donsker's theorem (see e.g. [2], pages 146 and 154).

## 2.2 The Demand Process

### 2.2.1 The Markov Model

Assume the arrivals form a Poisson flow with rate $\lambda$, and each of the (potentially infinitely many) elevators serves its passenger and comes back in an exponentially distributed random time with parameter $\mu$.

Then, we have an $M|M|\infty$ queue, which is a well studied object (see e.g. [1], page 103). In particular, it is known that the steady-state distribution of the number of elevators in use is Poisson with parameter $\lambda/\mu$.

This result is in excellent agreement with the simulation of Section 5.1.1, despite the non-exponential service times there.

### 2.2.2 One-Floor Traffic: Deterministic Service Time

We now turn to studying the case when the service time is not exponential. But first, for simplicity, we look at the case of only one floor. This means that the "service time" (i.e. the round-trip time for the elevator) is a known fixed (deterministic) positive number.

In other words, we are considering an $M|D|\infty$ queue.

Let's focus our attention only at the passengers going to one particular floor, say the $i^{\text{th}}$ floor. Let $N_t^{(i)}$ denote the number of passengers to the $i^{\text{th}}$ floor that arrived to the lobby by time $t$. We will assume that it is a Poisson arrival process with rate $\lambda_i$. Let $D_i$ be the return-trip time to the $i^{\text{th}}$ floor and back.

Then, for $t > D_i$, we know that $N_{t-D_i}^{(i)}$ will be equal exactly to the number of passengers to the $i^{\text{th}}$ floor that arrived, were served, *and* whose elevator has *returned back* to the lobby by time $t$. Therefore, the difference $N_t^{(i)} - N_{t-D_i}^{(i)}$ will show the number of elevators (to the $i^{\text{th}}$ floor) in use at time $t$.

For $t > D_i$, the elevator demand generated by the $i^{\text{th}}$ floor traffic is

$$\Delta_t^{(i)} := N_t^{(i)} - N_{t-D_i}^{(i)}.$$

For each fixed $t > D_i$, the r.v. $\Delta_t^{(i)}$ is a Poisson random variable with parameter $\lambda_i D_i$.

Let's take a closer look at the process $\Delta^{(i)}$. It follows from the properties of Poisson processes, that $\Delta^{(i)}$ is a stationary process. This means that, given any positive times $D_i \le t_1 < t_2 < \cdots < t_m$, and an $h > 0$, the random vectors

$$(\Delta_{t_1}^{(i)}, \Delta_{t_2}^{(i)}, \ldots, \Delta_{t_m}^{(i)}) \qquad \text{and} \qquad (\Delta_{t_1+h}^{(i)}, \Delta_{t_2+h}^{(i)}, \ldots, \Delta_{t_m+h}^{(i)})$$

have the same distribution.

The process $\Delta^{(i)}$ has a constant mean

$$m^{(i)}(t) := \mathbf{E}\Delta_t^{(i)} = \lambda_i D_i,$$

and a covariance function

$$K^{(i)}(s,t) := \operatorname{cov}(\Delta_s^{(i)}, \Delta_t^{(i)}) = \mathbf{E}(\Delta_s^{(i)} - \lambda_i D_i)(\Delta_t^{(i)} - \lambda_i D_i) = \lambda_i (D_i - |t-s|)^+,$$

where we have used the notation for positive part:

$$x^+ := \begin{cases} x, & \text{if } x \geq 0, \\ 0, & \text{if } x < 0. \end{cases}$$

In particular, if $|t-s| > D_i$, then $\Delta_s^{(i)}$ and $\Delta_t^{(i)}$ are uncorrelated. In fact, we can say more. Since Poisson processes have independent increments, it follows that $\Delta_s^{(i)}$ and $\Delta_t^{(i)}$ are *independent* whenever $|t-s| > D_i$.

Also, from (1) it follows that if $\lambda_i D_i$ is relatively large, then the process

$$\frac{\Delta_t^{(i)} - \lambda_i D_i}{\sqrt{\lambda_i D_i}} = \frac{N_t^{(i)} - N_{t-D_i}^{(i)} - \lambda_i D_i}{\sqrt{\lambda_i D_i}}$$

is well approximated in distribution by a process of the form

$$W_t - W_{t-D_i},$$

where $W_t$ is a standard Wiener process.

### 2.2.3 Combining All Floors

Finally, we turn to the full-fledged $M|G|\infty$ queue model and look at all floors now. The flow of all passengers is a union of Poisson processes, like the one described above.

It follows from the discussion in Section 2.2.2 that, for $t$ large enough (more precisely, for $t > \max_i D_i$) the elevator demand at time $t$ is given by

$$\Delta_t = \sum_i \Delta_t^{(i)} = \sum_i \left( N_t^{(i)} - N_{t-D_i}^{(i)} \right).$$

Assume that all floor arrivals are independent. Then, for each fixed $t$ (large enough), the demand is a Poisson random variable with parameter $\sum_i \lambda_i D_i$.

Based on the discussion of the one-floor case we can say that, for times $t > \max_i D_i$, the process $\Delta$ is stationary, and that $\Delta_s$ and $\Delta_t$ are *independent* whenever $|t-s| > \max_i D_i$.

For the mean and covariance functions we have:

$$\begin{aligned} m(t) &:= \mathbf{E}\Delta_t = \sum_i \lambda_i D_i, \\ K(s,t) &:= \operatorname{cov}(\Delta_s, \Delta_t) = \sum_i \left[ \lambda_i (D_i - |t-s|)^+ \right]. \end{aligned}$$

Observe again that $m(t)$ is constant (does not depend on $t$) and that $K(s,t) = 0$ whenever $|t - s| > \max_i D_i$.

If all $\lambda_i D_i$ are relatively large, then the demand at any fixed time $t$ is approximately normal, with mean and variance both equal to $\sum_i \lambda_i D_i$.

Again, all these results agree perfectly with the results of the simulations in section 5.1.1.

## 3 Analysis of Clustering

In this section, as before, we deal exclusively with uppeak traffic.

### 3.1 Clustering of Passengers

The previous section gives us a picture of the demand process. Since in reality we are dealing with only finitely many elevators, we will not be able, in general, to provide every passenger with a separate elevator. In those instances when the demand exceeds the number of elevators available, we will need to put some passengers together, so that they share elevators. Thus, given the flow of passengers, we must divide it into separate clusters. The number of clusters at any moment should be less than or equal to the number of elevators available.

How should we do the clustering? One simple approach is to let the elevator wait in the lobby (some "dwell time"), so that more passengers can get in. Since passengers arrive at random times and go to different floors, this form of clustering is hardly optimal.

Here is the question. If we know in advance the exact times when the passengers will arrive, as well as their destinations, can we divide the flow of customers in an optimal way (of course we can, at least theoretically), and how to do this efficiently (enumerating all the possibilities and comparing them is not a feasible approach)?

Of course, if we want to optimize, we must have an objective. In this section we will choose the average waiting time as our measure of performance. We will try to minimize it. (Another possible criterion is the maximum waiting time, which we would also minimize; however we don't look at this one now.)

### 3.2 Marginal Reduction of Elevators, and a Counterexample

With the intent to simplify the clustering process, the following procedure was proposed initially. Look at the points in time where the demand is maximal. The times when each of these maxima is first reached are arrival times of some passengers. Combine each of these passengers with another passenger, which will reduce the maximal demand by 1. Since the passengers that we need to combine with someone are few, then the optimal clustering (at this level) is computationally cheap. Once we have done so, we continue inductively, one step at a time, optimizing at each step, until the maximal demand reduces to the number of elevators available. This will be much easier and computationally tractable than trying to do all the clustering at once. The hope behind this stepwise (marginal) reduction is that, if we optimize at each step (relative to what we have obtained on the previous step), the combined steps will be optimal, too.

All these hopes are shattered by the following counterexample.

Assume that there are three arrivals: passenger $p_1$ arrives at time $t_1 = 0$ and requests the 1st floor, passenger $p_2$ arrives at time $t_2 = 1$ and requests the 30th floor, and passenger $p_3$ arrives at time $t_3 = 3$ and requests the 1st floor. We can specify the arrival sequence as follows:

$$S = \{p_1, p_2, p_3\} = \{(0, 1), (1, 30), (3, 1)\}\,. \tag{2}$$

The return trip durations are 4.1, 65, and 4.1 seconds, respectively. The maximal demand is 3.

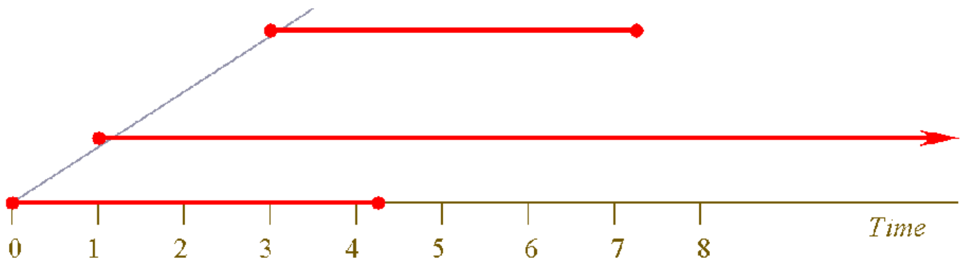

Figure 1: A graphical representation of the case used in the counterexample.

The optimal assignment for the 2 elevators

$$\{\{E_1;\ p_1, p_2\}, \{E_2;\ p_3\}\} \tag{3}$$

produces a wait-time vector (1, 0, 0). The reduced sequence is $S_1 = \{p_1\&p_2, p_3\}$. The $S_1$-optimal assignment

$$\{\{E_1;\ p_1\&p_2, p_3\}\} \tag{4}$$

produces a wait-time vector $(3, 2, 0)$. The optimal assignment for a single elevator,

$$\{\{E_1;\ p_1\}, \{E_1;\ p_2, p_3\}\} \tag{5}$$

produces a wait-time vector $(0, 3.1, 1.1)$. The latter assignment produces a better average wait-time output than the above $S_1$-optimal assignment.

At the time of writing, it is not yet clear whether the reduction fails under the maximum wait-time metric.

## 3.3 Cluster Algebra

In this section we will formalize the idea of a cluster, and we will introduce an algebraic operation on these formal clusters. This operation corresponds to putting the two clusters together into one new cluster.

The simplest cluster consists of just one passenger. Once several passengers have been united into a cluster, we can think of this cluster as a "generalized passenger". The advantage of this formalism is that we will be able to treat clusters and passengers the same way in our aggregation procedures.

We consider a building with a lobby (where all passengers get in), and $N$ destination floors on which passengers can get out.

A cluster, which is a set of passengers, will be formalized as an ordered triple $p = (t, w, F)$, where

- $t$ is the time of arrival of last passenger in the cluster
- $w$ is the sum of how long each of the passengers waits for the last one.
- $F$ is a vector of size $N$; For $k = 1, \ldots, N$, the component $F(k)$ denotes the number of passengers in the cluster getting out at floor $k$.

Strictly speaking, the cluster should also keep track of which passengers comprise it. However, we will prefer to keep this information separately (e.g., a list of all the passengers with assignments to their respective clusters).

Given a cluster $p = (t, w, F)$ of passengers, we can identify or compute a variety of quantities.

| Quantity | Description | Formula |
|---|---|---|
| $T(p)$ | Time of arrival of last passenger | $t$ |
| $W(p)$ | Total waiting time | $w$ |
| $NS(p)$ | Number of stops | $\sum_{k=1}^{N} \mathbf{1}\{F(k) \neq 0\}$ |
| $NP(p)$ | Number of passengers | $\sum_{k=1}^{N} F(k)$ |
| $H(p)$ | Highest floor for this cluster | $\max\{k : F(k) \neq 0\}$ |
| $AW(p)$ | Average waiting time | $W(p)/NP(p)$ |
| $D(p)$ | Return trip time | a function of $F$ |

Table 1: Quantities related to a cluster of passengers

It should be noted that the quantity $D(p)$, the return-trip duration for the particular cluster, can be computed uniquely based on the vector $F$. The concrete formula (or algorithm) may vary with the model.

For example, if we assume that the elevator travels at speed 2.5 seconds per floor, that it takes 5 seconds to open or close the doors, and that each passenger takes 3 seconds to get in or out, we get the following formula:

$$D(p) = (2)(2.5)H(p) + (2)(5)NS(p) + (2)(3)NP(p). \tag{6}$$

This is a rather simplistic formula, which does not account for the fact that, if there is a stretch of floors with no stops, the elevator moves much faster than when the stops are often. More sophisticated formulas may be needed for $D(p)$ (see, e.g. [4]).

Two things should be clear, though. First, the quantity $D(p)$ can always be determined from $F(p)$, by a formula or algorithm. Second, the concrete form of this formula is irrelevant for our further discussion in this section. We will need only a couple of properties of $D(p)$ which we will postulate a little further below.

We now proceed to define an operation on clusters. Let $p_1 = (t_1, w_1, F_1)$ and $p_2 = (t_2, w_2, F_2)$ be two clusters without common passengers. We define the *sum* of these clusters to be another cluster $p = (t, w, F)$, where

$$\begin{aligned} t &= \max\{t_1, t_2\} \\ w &= \begin{cases} w_1 + w_2 + NP(p_1)(t_2 - t_1), & \text{if } t_1 \le t_2, \\ w_1 + w_2 + NP(p_2)(t_1 - t_2), & \text{if } t_2 \le t_1, \end{cases} \\ F &= F_1 + F_2. \end{aligned}$$

We denote this operation as

$$p = p_1 \oplus p_2.$$

The following properties are easy to verify:

$$\begin{aligned} p_1 \oplus p_2 &= p_2 \oplus p_1 \\ (p_1 \oplus p_2) \oplus p_3 &= p_1 \oplus (p_2 \oplus p_3) \\ NP(p_1 \oplus p_2) &= NP(p_1) + NP(p_2) \\ W(p_1 \oplus p_2) &\ge W(p_1) + W(p_2) \\ H(p_1 \oplus p_2) &= \max\{H(p_1), H(p_2)\} \\ NS(p_1 \oplus p_2) &\le NS(p_1) + NP(p_2) \\ NS(p_1 \oplus p_2) &\ge \max\{NS(p_1), NS(p_2)\}. \end{aligned}$$

We also postulate the following natural properties of $D$:

$$D(p_1 \oplus p_2) < D(p_1) + D(p_2), \tag{7}$$
$$D(p_1 \oplus p_2) > \max\{D(p_1), D(p_2)\}. \tag{8}$$

These are properties that agree with our intuition. Any formula or algorithm that purports to compute $D(p)$ in a realistic way should satisfy these two properties. For example, the formula (6) does.

As a corollary of the second inequality, we get

$$NP(p_1 \oplus p_2)D(p_1 \oplus p_2) > NP(p_1)D(p_1) + NP(p_2)D(p_2) \tag{9}$$

which will be useful in next subsection.

The following example will clarify the notions introduced in this subsection. Consider a building with a lobby and $N = 8$ floors. Six passengers arrive as shown in Table 2.

Let's cluster the six passengers as follows:

- cluster $p_1$: passenger 2;
- cluster $p_2$: passengers 1, 4, 5;
- cluster $p_3$: passengers 3, 6.

| Passenger No. | Arrival Time | Going to Floor |
|---|---|---|
| 1 | 0 | 3 |
| 2 | 1.8 | 5 |
| 3 | 5.3 | 7 |
| 4 | 47.6 | 4 |
| 5 | 50.8 | 4 |
| 6 | 71.7 | 2 |

Table 2: Six passengers to eight floors

We then have:

$$\begin{aligned} p_1 &= (1.8, 0, [0,0,0,0,1,0,0,0]) \\ p_2 &= (50.8, 54, [0,0,1,2,0,0,0,0]) \\ p_3 &= (71.7, 66.4, [0,1,0,0,0,0,1,0]) \end{aligned}$$

For these clusters we have:

$$\begin{array}{lll} T(p_1) = 1.8, & T(p_2) = 50.8, & T(p_3) = 71.7, \\ W(p_1) = 0, & W(p_2) = 54, & W(p_3) = 66.4, \\ NS(p_1) = 1, & NS(p_2) = 2, & NS(p_3) = 2, \\ NP(p_1) = 1, & NP(p_2) = 3, & NP(p_3) = 2, \\ H(p_1) = 5, & H(p_2) = 4, & H(p_3) = 7, \\ AW(p_1) = 0, & AW(p_2) = 18, & AW(p_3) = 33.2. \end{array}$$

If we decide to combine clusters $p_1$ and $p_3$ into one, we get a new cluster

$$p_4 = p_1 \oplus p_3 = (71.7, 136.3, [0,1,0,0,1,0,1,0]).$$

## 3.4 Metric-Based Clustering

The question now is, given all the arrival times and floor destinations of all the passengers, how do we divide them into clusters?

While an optimal clustering might be difficult to achieve, we can hope to find a near-optimal clustering based on the following heuristic idea.

When two customers are combined, there are two sources of delay. First, one of the passengers must wait for the other to arrive. Second, because of the extra passenger in the car, the round-trip time will increase. This means that the elevator will be unavailable for a longer period of time, and that one of the passengers will have to travel longer than if he was alone.

Thus, it becomes clear that we would like to put together customers which are "close" to each other in two ways: they arrive at close times, and/or they go to the same floor (or to close floors).

We would like define a "metric" ("distance") between passengers by some formula which takes into account these two factors. Bearing in mind the benefits of the "cluster algebra" developed in the previous subsection, it would be good to be able to extend this metric to distances between clusters (and not just between individual passengers).

A good candidate for a distance between two clusters, $p_1$ and $p_2$ is the following:

$$\begin{aligned}\delta(p_1, p_2) = & [W(p_1 \oplus p_2) - W(p_1) - W(p_2)] \\ & + c\,[NP(p_1 \oplus p_2)D(p_1 \oplus p_2) - NP(p_1)D(p_1) - NP(p_2)D(p_2)]\,,\end{aligned} \tag{10}$$

where $c$ is a positive "weight". Observe that, because of (9), the quantity $\delta(p_1, p_2)$ is always positive. It is also symmetric, i.e. $\delta(p_1, p_2) = \delta(p_2, p_1)$.

It should be noted, however that it does not satisfy, in general, the triangle inequality; in other words,

$$\delta(p_1, p_3) \leq \delta(p_1, p_2) + \delta(p_2, p_3)$$

is not always true. A counterexample can be easily constructed, if we take the three clusters going to the same floor, and let $t_2 < t_1 < t_3$ be very close, and take $NP(p_2) = NP(p_3) = 1$, while letting $NP(p_1)$ be huge.

Unpleasant as this is, we can still use this "distance"; we just need to always be cautious and remember that $\delta$ is not a real metric.

Based on this distance, we can suggest the following clustering algorithm. Compute all the distances between the individual passengers. Cluster the two that are closer. Replace these two passengers with the new cluster, and thus reduce the size of the problem by 1. Repeat this over and over, until the demand function for the number of clusters requiring service at any given time no longer exceeds the number of elevators available.

It is important to understand that the algorithm does not aim at clustering until the number of all clusters is equal to the number of elevators. Instead, it looks at the demand function (=number of clusters arrived minus number of clusters served, at moment $t$), and clusters until the demand at any moment in time does not exceed the number of elevators. (Because, if it does, we will be forced to cluster further anyway.) In effect, this is Slaven's algorithm, just the criterion for putting clusters together is different.

Even so, there is a major gap in this approach. It does not allow for one cluster waiting as a whole for an elevator. The example in Subsection 3.2 works again to illustrate what happens. If we have just one elevator, any distance-based clustering will want to cluster all 3 passengers together. The optimal clustering requires to have two clusters, with the second waiting a while for the elevator to become available.

There is a rough idea how to deal with this. We need to keep track of how much "overlap" there exists.

All in all, we don't believe that even the corrected algorithm will give an optimal clustering, but just a "near-optimal" one, whatever this means.

More research is required.

### 3.4.1 Clustering Code

A Matlab code was developed that implemented this algorithm (main program "elevator.m") that was distributed to the team members.

## 4 Formulation of *a posteriori* uppeak mixed integer programming problem

Consider a building with $M$ elevators, each capable of accommodating $P_c$ passengers. Suppose we are given the arrival times $t_i$ and destination floors $f_i$ of $P$ passengers arriving in the lobby of the building during the uppeak period. The problem is to determine a schedule for the elevators and an assignment of passengers to elevators so as to minimize either the average passenger waiting time or the maximum passenger waiting time. (Waiting time is defined as the period between a passenger's arrival in the lobby and the departure time of his/her elevator.)

Let

$$x_{i,j,k} = \begin{cases} 1 & \text{if passenger } i \text{ is assigned to trip } \ j \text{ of elevator } k \\ 0 & \text{otherwise} \end{cases}$$

and let $\tau_{j,k}$ denote the starting time of the $j^{\text{th}}$ trip of elevator $k$. The waiting time of passenger $i$ is given by

$$\sum_{j=1}^{P}\sum_{k=1}^{M} x_{i,j,k}\left(\tau_{j,k} - t_i\right).$$

(We use $P$ as the upper limit of the first sum because the number of trips an elevator makes cannot exceed the number of passengers.) Therefore, the average passenger waiting time is

$$AW = \frac{1}{P}\sum_{i=1}^{P}\sum_{j=1}^{P}\sum_{k=1}^{M} x_{i,j,k}\left(\tau_{j,k} - t_i\right)$$

and the maximum passenger waiting time is

$$MW = \max_{1\le i\le P}\sum_{j=1}^{P}\sum_{k=1}^{M} x_{i,j,k}\left(\tau_{j,k} - t_i\right).$$

We formulate the assignment problem as follows:

Minimize $AW$ (or $MW$)
subject to the constraints

$$\begin{cases} \displaystyle\sum_{j=1}^{P}\sum_{k=1}^{M} x_{i,j,k} = 1, 1 \le i \le P & \text{(each passenger takes exactly one trip)} \\ \displaystyle\sum_{i=1}^{P} x_{i,j,k} \le P_c, 1 \le j \le P,\ 1 \le k \le M & \text{(no more than } P_c \text{ passengers per trip)} \\ \text{and} & \\ \tau_{j+1,k} \ge \tau_{j,k} + D_{j,k} & \text{(elevator can't start trip before end of} \\ & \text{previous trip).} \end{cases}$$

Here $D_{j,k}$ denotes the round trip time of trip $j$ of elevator $k$. $D_{j,k}$ is a specified function of the number of passengers $\left(n_{j,k} = \sum_{i=1}^{P} x_{i,j,k}\right)$ on this trip and of the highest floor requested $\left(H_{j,k} = \max_{1 \le i \le P} x_{i,j,k} f_i\right)$. For example, if the elevator has a flight time of $v$ seconds per floor and if it takes each passenger time $t_s$ to get on the elevator and $t_s$ to get off, we might have
$D_{j,k} = 2n_{j,k}t_s + 2vH_{j,k}$.

This model was formulated during the workshop; no computations were made with this model. We now move on to Monte Carlo simulations performed during the workshop.

## 5 Simulations of uppeak traffic

Discrete event simulations of up-peak elevator traffic take a sequence of passenger arrival times and then assign responding elevators to the passengers according to a pre-defined algorithm. When the passenger inter-arrival times and destination floors are randomly selected according to some probability distributions, the resulting Monte Carlo simulations can be used to statistically test the response of the elevator algorithm to a wide range of "typical" passenger arrival scenarios that are consistent with the specified floor and arrival time distributions.

We approach the problem of simulating and analyzing up-peak elevator traffic in several stages given in the following subsections. In this way we will be able to separate out the factors that influence the average number of elevators in use and the average waiting time per passenger: the mean passenger arrival rate, the elevator dwell time, the maximum number of elevators, and the passenger capacity per elevator.

### 5.1 Unlimited elevators: As many elevators as needed

For the first model problem, we were influenced by the recent film *The Matrix Reloaded.* We envisioned a building as possible in the movie, where as many elevators as needed can be added to the building so that no passengers have to wait in the lobby.

A code was written in C that implemented the following conceptual framework; a detailed algorithm is given in the appendix. At each passenger arrival, indexed by $n = 1, 2, 3, \cdots$, an elevator was provided to take the passenger to floor $f$, which was randomly generated from a discrete distribution on the floors $[1, 2, \cdots, N]$ (a uniform distribution was used as a default). The round-trip duration time is given by

$$D = 2vf \tag{11}$$

where $v$ is the flight time (i.e., time required in seconds to ascend or descend one floor) and $f$ is the destination floor. The flight time for ascent or descent may be different for uppeak operation, but they were assumed constant and equal for this model. The arrival of passengers was assumed to be a Poisson process with mean arrival rate $\lambda$ (in calls per

| $n$ | $N$ | $v$ | $\lambda$ | $E$ | $\sigma$ |
|---|---|---|---|---|---|
| 9,000 | 50 | 1 | 1 | 50 | $10/\sqrt{2}$ |
| 250,000 | 50 | 1 | 1 | 50 | $10/\sqrt{2}$ |
| 9,000 | 50 | 1 | 2 | 100 | 10 |
| 250,000 | 50 | 1 | 1/2 | 25 | 5 |
| 250,000 | 100 | 1 | 1 | 100 | 10 |
| 250,000 | 100 | 2 | 1 | 200 | $10\sqrt{2}$ |

Table 3: The mean $\mu$ and the standard deviation $\sigma$ for the *Matrix Reloaded* model when elevators are created on demand if all previous elevators are in use. The distribution of elevators required for large $n$ is normal with mean and standard deviation $(E, \sqrt{E})$ with $E = \lambda v(N+1)$.

second), where the probability $p$ of $n$ calls being registered in the time interval $T$ is given by

$$p(n) = \frac{(\lambda T)^n}{n!} e^{-\lambda T}. \tag{12}$$

The average number of passenger arrivals in the time interval $T$ is $\lambda T$.

### 5.1.1 Zero waiting time: No clustering of passengers

The simplest version of the the up-peak problem with an unlimited number of elevators specifies that each arriving passenger gets their own private elevator. The elevator will leave instantly after the passenger enters, hence there is no waiting time. After the passenger arrives at their desired floor, the elevator will return to the lobby to be re-used.

In this case, the only parameters are the passenger arrival rate $\lambda$ (Poisson process parameter), the height of the building (number of floors above the lobby) $N$ and the flight time $v$ in seconds/floor.

The results of the simulation for this case serving $10^5$ passengers is shown in Figure 2. The number of elevators required at any given time (or passenger number $n$) is shown by the jagged Gray curve. One can see that the peak demand, shown by the solid (piecewise constant) curve, increases slowly for large $n$ because of increasingly-rare events (i.e. cases when none of the existing elevators will return to the lobby in time to pick-up the next passenger, then a new elevator must be added). For small $n$ (for the first 50 or so passengers), the peak demand equaled the instantaneous number of elevators in use, however these numbers later separate when returning elevators become available in the lobby.

The number of elevators in use may be described statistically, see Figure 3. The distribution of elevators required is found to be static for large enough $n$. The mean of the distribution is found to be $E = \lambda v(N + 1)$; the variance turns out to the same, see Table 5.1.1. This indicates that the number of elevators in use is a Poisson process, as is the process for passenger arrival. Because $n$ is large, the Poisson process appears to have a normal distribution; more discussion of Poisson processes appears in Section 2.1.3.

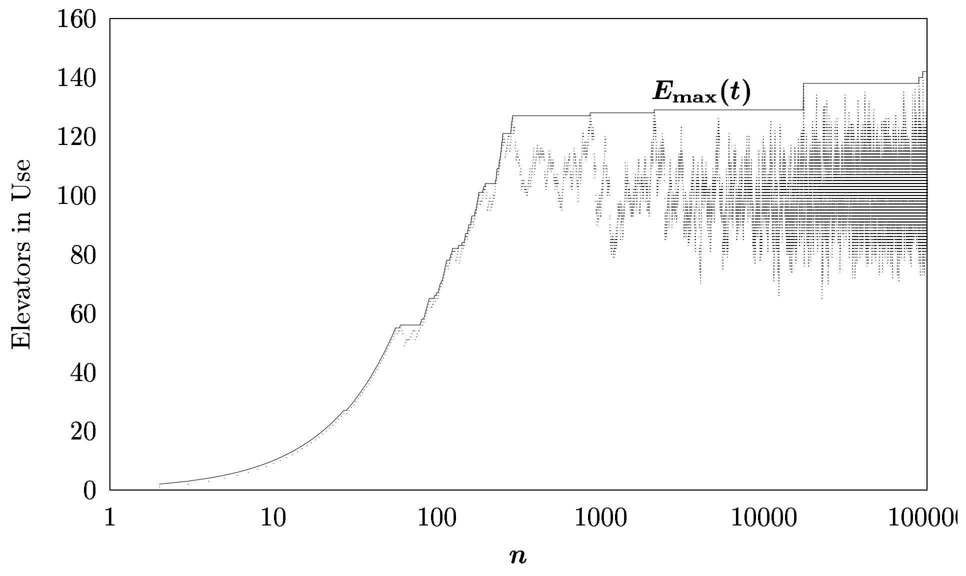

Figure 2: Results for the *Matrix Reloaded* model; here $\lambda = 1$, $v = 1$ and $N = 100$. The peak demand of elevators $E_{\max}(t)$ *vs.* passenger number $n$ is given by the red curve. The instantaneous number of elevators in use at any given time is the jagged gray curve.

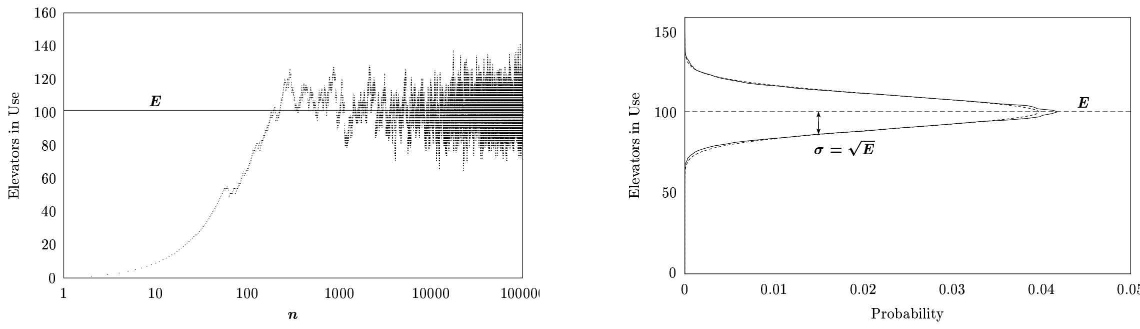

Figure 3: On the left is the evolution of the number of elevators in use; on the right is the probability density function for the number of elevators in use $n > 1000$. The distribution appears to be close to a normal with average $E = \lambda v(N + 1)$ and variance $E$.

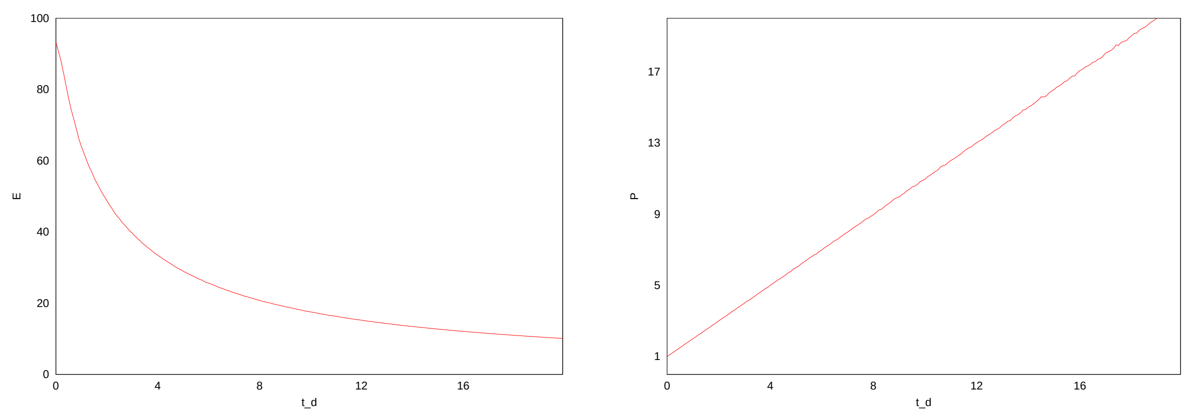

Figure 4: The influence of including nonzero dwell time $t_d$ in the simulation. On the left is the average number of elevators in use $E$ during a simulation as a function of dwell time; on the right is the average number of passengers $P$ in each elevator.

### 5.1.2 Clustering is good: the effect of dwell time

If one were not supplying an elevator for each passenger, some passengers would have to wait and this would directly impact the number of elevators needed. In this case, a dwell time $t_d$ (measured from the time that the first passenger enters an elevator) is included in the simulation. This delay in the elevator departure from the lobby allows passengers to collect in the elevator and thus reduce the demand for elevators. In the simulation, arrival of passengers is still a Poisson process with parameter $\lambda$; passengers arrive individually with an average interval of $1/\lambda$ between arrivals and an exponential distribution of inter-arrival time intervals. Elevators depart after $t_d$ time units after the entry of the first passenger no matter how many or how few passengers are on the elevator; these elevators have no passenger capacity limit. If all elevators are in use (i.e. away from the lobby) when the next passenger arrives, then a new elevator is created.

Increasing dwell time reduces the demand for elevators dramatically. In Figure 4, the average demand for elevators is shown as a function of time. The average number of passengers per elevator clearly increases with dwell time as well; the average number of passengers is given by $P = 1 + \lambda t_d$. It is clear that the dwell time has an important impact on the number of elevators required to handle uppeak traffic.

The average waiting time as a function of $t_d$ is shown in Figure 5. The waiting time increases less rapidly than linearly with small $t_d$, but becomes linear with larger $t_d$. The slope of the line is half that of the increase of the number of passengers per unit time ($\lambda$); this is because the average wait time of the passengers before the elevator departs is about half of $t_d$.

Increasing the dwell time also causes the variance of the number of elevators in use to be decreased (not shown graphically). The distribution of elevators is then no longer Poisson, as can be seen because the mean and variance will now differ substantially in general (the variance is typically much smaller).

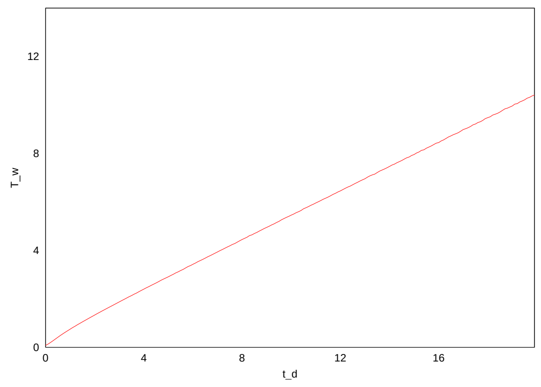

Figure 5: The average waiting time $t_w$ (before the elevator departs) is plotted against $t_d$; it is linear but the slope is half that of the increase in the number of passengers with $t_d$. Here $\lambda = 1$, $v = 1$ and $N = 100$.

## 5.2 Closer to reality: A finite number of elevators

In this section, the effect of having a limited number of elevators is studied; the *Matrix Reloaded* aspect of adding (turning on) as many elevators as needed is now constrained by the fixed finite maximum number of elevators available in a realistic building. Let $M$ denote the total number of elevators available.

The average waiting time as a function of the dwell time $t_d$ with a finite number of elevators is shown in Figure 6. The waiting time decreases to a minimum at about $t_d = 2$ for this set of parameters, and increases more rapidly than linearly for larger $t_d$.

Increasing dwell time still reduces the demand for elevators dramatically, but in a perhaps unexpected way. In Figure 7, the average demand for elevators is shown as a function of time. The number of elevators in use (dashed curve) is the maximum $M = 30$ (dotted curve) for $t_d = 0$ and it decreases slowly for small $t_d$. However, for $t_d > 7$, the number of elevators in use is practically identical to the result for the *Matrix Reloaded* case (solid curve) with a nonzero dwell time. The average number of passengers per elevator clearly increases with dwell time as well; the average number of passengers is still given by $P = 1 + \lambda t_d$, which is identical to the previous case. It is clear that the dwell time still has an important impact on the number of elevators required to handle uppeak traffic, but beyond a critical value for the dwell time, in this case $t_d > 7$, the maximum number of elevators has no effect.

## 5.3 Closer still: A finite number of limited capacity elevators

In this case, the number of elevators does not exceed $M$. The elevator remains at the lobby for the dwell time $t_d$ or until the elevator reaches its capacity $P_c$, whichever comes first. The dwell time or the time to reach capacity begins with a single passenger already in the elevator.

The results for this case are summarized in Figures 8 through 9. The average demand for this case is shown in Figure 8; the number of passengers transported is limited by the

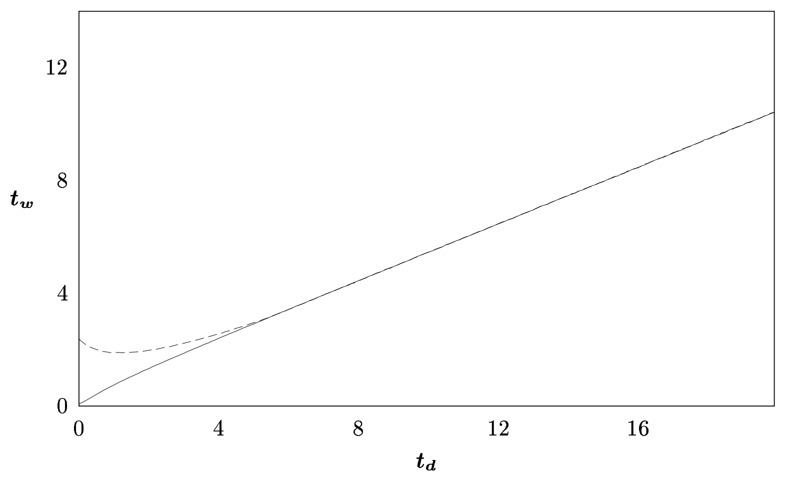

Figure 6: The average wait time $t_w$ (before the elevator departs) is plotted against $t_d$; the solid curve is from Section 5.1.1 while the dashed curve are results with a maximum number of elevators $M$ available. Here $\lambda = 1$, $v = 1$, $N = 100$ and $M = 30$.

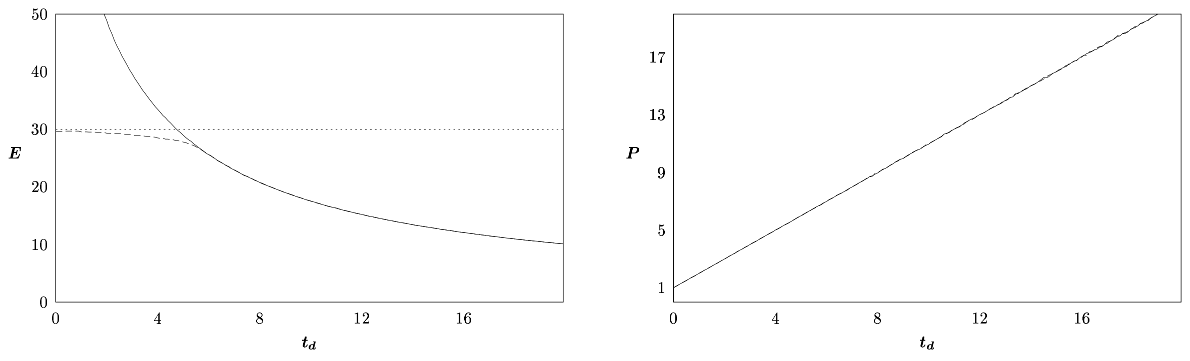

Figure 7: Results including nonzero dwell time $t_d$ and a maximum number of elevators in the simulation. On the left is the average number of elevators in use $E$ during a simulation as a function of dwell time; on the right is the average number of passengers $P$ in each elevator. The solid curve on the left is the case with an unlimited number of elevators; the dashed curve is the number of elevators in use with a maximum number $M$ available. Here $\lambda = 1$, $v = 1$, $N = 100$ and $M = 30$.

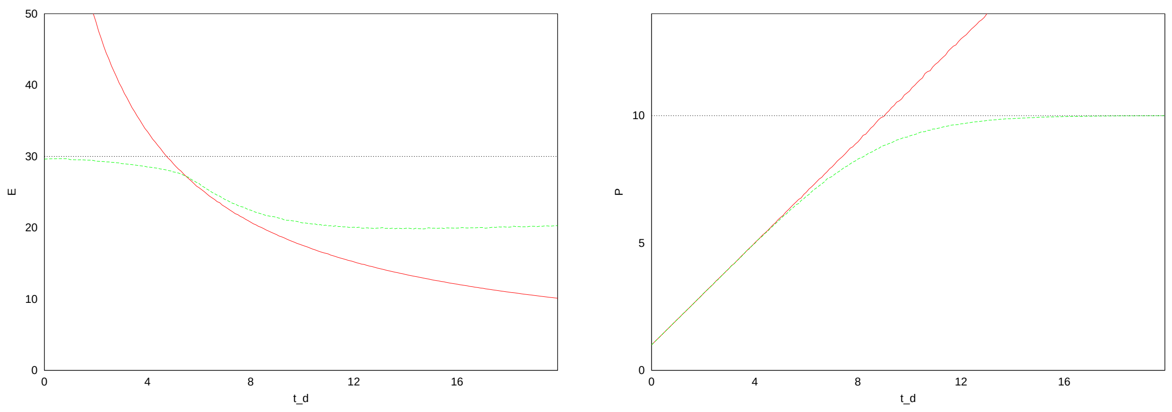

Figure 8: Results from simulations including nonzero dwell time $t_d$, finite capacity $P_c$ and a finite number of elevators $M$. Here $\lambda = 1$, $v = 1$ and $N = 100$. (Left) The average number of elevators in use $E$ during a simulation is plotted as a function of dwell time. The dashed curve is the result of the current model; the solid curve is the case of section 5.1.2. (Right) The average number of passengers $P$ in each elevator is plotted against $t_d$. The dashed curve is the current model; the solid curve is the result of the simulation of section 5.1.2. The black horizontal line is the elevator capacity. As the dwell time increases, the green curve shows that the limiting behavior switches from the number of elevators to the capacity of each elevator.

number of elevators $M$ at small $t_d$, while for larger $t_d$, it is the capacity of the elevators that affects when they depart. There is a minimum in the number of elevators required at about $t_d \approx 14$ for this case.

Figure 8 shows that the number of passengers $P$ is not affected by the capacity for small $t_d$, which can be defined as $t_d < (P_c - 1)/\lambda$; this is because the delay time starts with one passenger in the elevator, and so $t_d$ is smaller than the average time required to fill the elevator. For large dwell time, $t_d > (P_c - 1)/\lambda$, then the elevator fills and departs before waiting a full dwell time. The number of passengers in the elevator then consistently becomes the capacity $P_c$.

The waiting time for this case is shown in Figure 9. The waiting time is minimized for a relatively small dwell time; for the small interval $4 \leq t_d \leq 6$, the behavior is hard to distinguish from the earlier model of Section 5.1.2 with no limit on the number of elevators or on the capacity of the elevators. Note that the $t_d$ which minimized the number of elevators does not correspond to any maximum or minimum in the average waiting time.

Waiting times and average numbers of elevators in use are shown in Figure 10. Figure 10 (Left) shows the expected waiting time *vs.* dwell time for case 4 for different numbers of elevators $M = 30, 25, 20, 15, 10, 5$. We can see the optimal dwell time increases as we decrease the number of elevators. In Figure 10 (Right), the average number of elevators in use is plotted against the dwell time for different numbers of elevators $M = 30, 25, 20, 15, 10, 5$. For smaller $M \leq 20$, we see that the elevators get filled up all

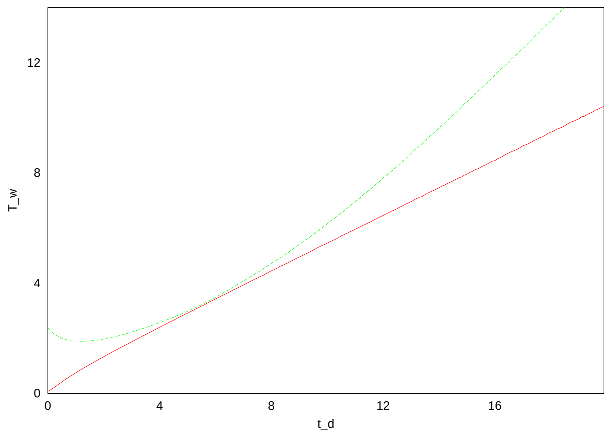

Figure 9: Further results from simulations including nonzero dwell time $t_d$, finite capacity $P_c$ and a finite number of elevators $M$ (again with $\lambda = 1$, $v = 1$ and $N = 100$). The average waiting time $t_w$ (before the elevator departs) is plotted against $t_d$. The dashed curve is the current model; the solid curve is the result of the simulation of section 5.1.2. A minimum waiting time appears in $t_d$ in the current case.

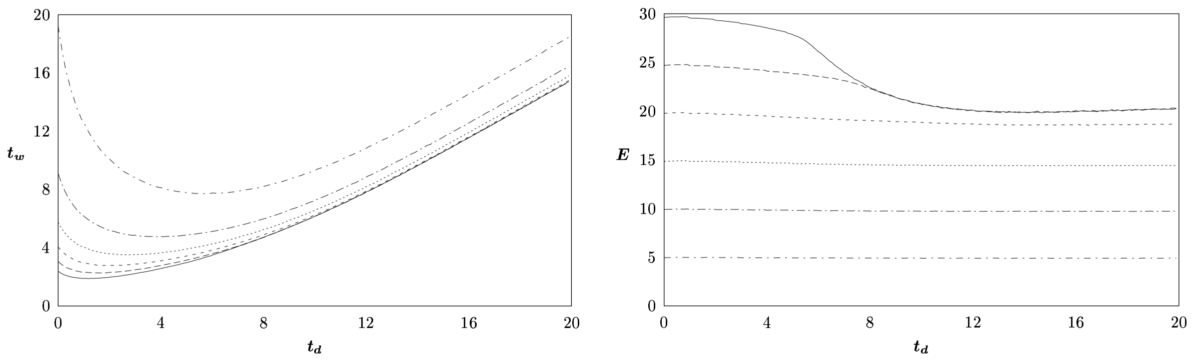

Figure 10: Further results from simulations including nonzero dwell time $t_d$, finite capacity $P_c$ and a finite number of elevators $M$ (still with $\lambda = 1$, $v = 1$ and $N = 100$). (Left) The average waiting time $t_w$ (before the elevator departs) is plotted against $t_d$ for different values of $M$; the solid curve is the result of the simulation of section 5.1.2. Here $M = 30, 25, 20, 15, 10, 5$ (from bottom to top). The optimal dwell time increases as we decrease the number of elevators. (Right) The average number of elevators in use $E$ during a simulation is plotted as a function of dwell time; here $\lambda = 1$, $v = 1$, $N = 100$; from the top down, $M = 30, 25, \cdots$. The solid curve is the case of section 5.1.2. The elevators apparently fill to capacity for $M \leq 20$ or so.

the time. We may be able to choose between overflowing crowds in the lobby and picking an optimal number of elevators to moderate crowding in the elevators. For large dwell times, the $M = 25$ and $M = 30$ elevator curves indicate that the elevators don't get any more crowded; this seems to suggest that 25, or maybe 24, elevators would be "good" for this set of simulated conditions.

## 5.4 Nonuniform probability distributions for passenger destination floor

Different probability density functions (PDFs) were used to examine the case where some floors are preferred destinations. If the mean of the distributions are the same, then the distribution of elevator demand has the same mean results at long time in the simulations.

## 5.5 Formulas for the mean demand for elevators

The mean demand for elevators at late time (with a stationary distribution) in the *Matrix Reloaded* case was found above to be $E = \lambda v(N+1)$. We now give a simple conservation of people argument to derive this result.

Let $\tau$ be the average round trip time of an elevator. If there are $E$ elevators, then on average an elevator becomes available in the lobby every $\tau/E$ time units, and so one passenger can be accommodated every $\tau/E$ time units. In order to balance this with the arriving passengers, we should equate this time with the average passenger arrival interval $1/\lambda$ (for a Poisson process); we obtain

$$\frac{\tau}{E} = \frac{1}{\lambda} \quad \text{or} \quad E = \lambda\tau. \tag{13}$$

We can obtain an expression for $\tau$ by using the same approximations as the previous ; in particular, we ignore the stopping time to obtain

$$\tau = 2vH. \tag{14}$$

Here $H$ is the expected value of the highest floor. For equal likelihood of stopping at any of the $N$ floors above the lobby, we have

$$H = \sum_{i=1}^{N} \frac{i}{N} = \frac{1}{N}\frac{N(N+1)}{2} = \frac{N+1}{2}. \tag{15}$$

Substitution yields

$$E = \lambda v(N+1). \tag{16}$$

This expression agrees very well with the results of the simulations in section 5.1.1. See Table 1.

We now want to consider the effects of dwell time and elevator capacity on the previously found We let the arrival rate of the passengers be $C$ (not necessarily a Poisson process), and let $P$ be the number of passengers in the car; we assumed one passenger per car before. To balance the average arrival of passengers with the average time for an elevator to become available:

$$\frac{\tau}{E} = \frac{P}{C} \quad \text{or} \quad E = \frac{C}{P}\tau. \tag{17}$$

Let the total return trip time be the sum of the dwell time and the duration

$$\tau = t_d + D \tag{18}$$

where

$$D = 2vH + (S+1)t_s \tag{19}$$

and $D$ is the elevator trip duration, $v$ is the flight time (time/floor), $t_s$ is the stop time for a single floor and $S$ is the expected number of stops. We now consider three different cases for the average demand for elevators. (We call these cases 2 - 4; case 1 is the single passenger *Matrix Reloaded* case already considered.)

Case 2 is for the so-called rectangular case of Ch. 5 of [4]. In this case, the number of passengers is assumed; in [4], the estimate $0.8Pc$ is used where $P_c$ is the capacity of the elevator. In this case, we have a uniform likelihood of selecting any of the $N$ floors, and so

$$\begin{aligned} P &= 0.8P_c & (20)\\ H &= N - \sum_{i=1}^{N}\left(\frac{i}{N}\right)^P \le N - \frac{1}{(P+1)N^{P-1}} & (21)\\ S &= N\left[1 - \left(\frac{N-1}{N}\right)^P\right] & (22)\\ E &= \frac{C}{P}\left[t_d + 2vH + (S+1)t_s\right] & (23) \end{aligned}$$

The formulas for $H$ and $S$ are derived in [4]. Note that for $P = 1$, $C = \lambda$, $t_d = t_s = 0$, we recover the first average demand formula 16. In this case $C$ is an input parameter.

Case 3 assumes a Poisson process with arrival rate $\lambda$ for the passengers with $t_d < (P_c - 1)/\lambda$. The expression $(P_c - 1)/\lambda$ is the time required, after the first passenger, to fill the elevator; this expression attempts to match the simulations since they begin timing after the first arrival. In this case, then, the dwell time is less than the average time required to fill the elevator. We then have

$$\begin{aligned} P &= 1 + \lambda t_d & (24)\\ C &= \lambda & (25)\\ H &= N - \sum_{i=1}^{N}\left(e^{-(1+\lambda t_d)/N}\right)^i = N - \frac{e^{-(1+\lambda t_d)/N}(1 - e^{-\lambda t_d})}{1 - e^{-(1+\lambda t_d)/N}} & (26)\\ S &= N\left(1 - e^{-(1+\lambda t_d)/N}\right) & (27)\\ E &= \frac{C}{P}\left[t_d + 2vH + (S+1)t_s\right]. & (28) \end{aligned}$$

| case | $t_d$ | $P_c$ | $N$ | $H$ | $E$ | $H_{sim}$ | $E_{sim}$ |
|---|---|---|---|---|---|---|---|
| 3 | 5 | 10 | 100 | 84 | 29 | 75 | 29 |
| 3 | 2 | 10 | 100 | 69 | 47 | 66 | 48 |
| 3 | 1/2 | 10 | 100 | 49 | 65 | 56 | 76 |
| 4 | 40 | 10 | 100 | 91 | 19 | 81 | 22 |
| 4 | 40 | 10 | 200 | 182 | 37 | 160 | 41 |
| 4 | 40 | 10 | 100 | 100 | 21 | 81 | 22 |
| 4 | 40 | 10 | 200 | 200 | 41 | 160 | 41 |

Table 4: Results from the formulas for the mean number of elevators (column 5) in use from cases 3 and 4 is compared to the simulation results (column 7). The expected highest floor $H$ is also given from each approach (columns 4 and 6). In each case, $t_s = 0$ and $\lambda = v = 1$. The first group for case 4 used the exact sum for the expected highest floor; the second group for case 4 used the approximate expected highest floor (rightmost term given in case 4).

Case 4 occurs when $t_d > (P_c - 1)/\lambda$; in this case, the elevator fills, on average, before the dwell time is complete. For a Poisson arrival process, the average elevator round trip time should contain $(P_c - 1)/\lambda$ rather than $t_d$, because in the simulations, the elevator departs when it is full. We then have $\tau = \frac{P_c-1}{\lambda} + 2vH + (S+1)t_s$. The mean elevator demand is then given by

$$P = P_c \tag{29}$$

$$C = \lambda \tag{30}$$

$$H = N - \sum_{i=1}^{N} \left(\frac{i}{N}\right)^{P_c} \leq N - \frac{1}{(P_c+1)N^{P_c-1}} \tag{31}$$

$$S = N\left[1 - \left(\frac{N-1}{N}\right)^{P_c}\right] \tag{32}$$

$$E = \frac{\lambda}{P_c}\left[t_d + 2vH + (S+1)t_s\right]. \tag{33}$$

We now compare the results of Cases 3 and 4 with simulations from section 5.3 for $t_s = 0$. The agreement with the simulations is fair; there is a definite problem with the expected highest floor comparison between the conservation arguments compared to the simulations. The formulas use the expected highest floor and number of stops from the end of Chapter 5 in [4]; these may need to be revisited and improved for the current use. In particular, one would hope to recover an expected highest floor result of $(N+1)/2$ as $t_d \to 0$, but this doesn't happen for the small $t_d$ case.

## 5.6 Compound Poisson arrival processes: times and group size

We considered the setup of a compound Poisson arrival process, but did not implement it during the workshop; the initial part of this discussion is after [3], p. 68. In this case we have a set of $k$ batches of passengers of size $i$ with arrival rate $\lambda^{(i)}$. Each stream of batches is a Poisson process. The total process has a Poisson event distribution of

$$\lambda = \sum_{i=1}^{k} \lambda^{(i)} \tag{34}$$

Any batch size $p$ then occurs with probability

$$f_b = \frac{\lambda^{(b)}}{\lambda}. \tag{35}$$

The probability of a transition from state $i$ to state $j$, $j > i$, during the time interval $(t, t + \Delta t)$ is approximately $\lambda^{(j-i)}\Delta t$; the probability of remaining at state $i$ is approximately $1 - \lambda\Delta t$. (Here the state is the batch size.) Consider batch sizes of 1, 2 or 3; the probability of batch size $i$ arriving is governed by the following system of differential-difference equations:

$$\begin{aligned}
\frac{dp_0}{dt} &= -\lambda p_0(t) && (36)\\
\frac{dp_1}{dt} &= \lambda^{(1)} p_0(t) - \lambda p_1(t) && (37)\\
\frac{dp_2}{dt} &= \lambda^{(2)} p_0(t) + \lambda^{(1)} p_1(t) - \lambda p_2(t) && (38)\\
\frac{dp_3}{dt} &= \lambda^{(3)} p_0(t) + \lambda^{(2)} p_1(t) + \lambda^{(1)} p_2(t) - \lambda p_3(t) && (39)\\
\frac{dp_n}{dt} &= \lambda^{(3)} p_{n-3}(t) + \lambda^{(2)} p_{n-2}(t) + \lambda^{(1)} p_{n-1}(t) - \lambda p_n(t),\ n \geq 3. && (40)
\end{aligned}$$

The probability density functions may be solved by using discrete transform methods ([3], p. 69 and his Appendix A). It is convenient to use the initial conditions $p_0(0) = 1$ and $p_i(0) = 0$, $i > 0$.

Typically one may solve the system using transform methods in order to explicitly solve for the probability density function of each batch size as a function of time. However, one may also discretize this set of ODEs directly and march it forward in time with the same time step as the simulation; this would be a convenient way to generate the probability of a given batch size at an arrival event without having to find an explicit solution. One could very easily plot the PDFs for the batch sizes if desired.

One could also create a compound Poisson process for the arrival of passengers where the number of passengers arriving in a given time interval is a sum of random batch sizes with a random number of batches. This possibility is discussed in [3] on p. 70-71. Bulk queues with Poisson batch arrivals are discussed in Ch. 10 of [3].

## 6 Summary

Progress was made in classifying how various approaches to elevator dispatching fit into queueing theory. And the rationale for the types of processes found in simulation was explained using probabilistic arguments.

A proposed algorithm to marginally reduce the number of elevators to arrive at an optimal solution to the average wait time was shown to have a counterexample. A cluster-based approach was developed in some detail, though more work remains to be done with that.

A mixed integer programming problem was formulated for minimizing the average or maximum wait time. No further work was done with this model due to the time constraints.

Simulation of elevator performance using a Monte Carlo approach gave insight into the number of elevators required for a variety of situations. The effects of dwell time, maximum available elevators and capacity of each elevator were studied in particular.

## References

[1] D. Gross and C.M. Harris. *Fundamentals of Queueing Theory.* (Wiley, New York, NY, 1998) p. 285, 303.

[2] P. Billingsley. *Convergence of Probability Measures.*(Wiley, 1999)

[3] Walter C. Giffin. *Queuing: Basic Theory and Applications.* (Grid, Columbus, OH, 1978).

[4] Gina Barney. *The Elevator Traffic Handbook: Theory and Practice.* (Spon, London, 2003).

## Appendix A: Algorithm for simulation

One approach to studying this problem was via Monte-Carlo simulations. That is direct numerical simulations of the expected elevator responses to a large random sampling of all possible passenger arrival scenarios. The results of these simulations are tabulated to yield probability distributions for the resulting properties of the elevator system from which we can obtain the expected values of the waiting time and other measures of efficiency.

```
Set system parameters
    assign $N$;                                        Number of floors in building, with PDF
    assign $M$;                                        Maximum number of elevators available
    assign $P_c$;                                      Maximum capacity per elevator
    assign $T_f$;                                      Elevator flight time per floor (inverse speed)
    assign $T_d$;                                      Elevator lobby dwell time
    assign $T_s$;                                      Elevator time per stop
    assign $\lambda$;                                  Passenger arrival rate - people/time
The first passenger arrives at time zero and is assigned to the first elevator
Passenger-based variables
    $I = 1$;                                           Passenger number
    $T_I = 0$;                                         Arrival time of $I^{\text{th}}$ passenger
    assign $F_I$ a random value in $[1 : N]$ according to a discrete PDF    Assign destination floor
```

```
Elevator-based variables
    $J = 1$;                                                    Current total elevator number
    $T_0[J] = T_I$;                                             Start elevator “mission timer”
    $F_{\max}[J] = F_I$;                                        Maximum floor requested
    $P[J] = 1$;                                                 Population of elevator
    $T_R[J] = T_0[J] + 2F_{\max}[J]T_f + P[J]T_s + T_d$;        Current expected return time to lobby
Main Loop for all other passengers
Passenger-based variables
    $I \to I + 1$;                                              Add a new passenger
    assign $\Delta T_I$ from an exponential random variable, mean $\lambda$;   Poisson Process arrivals
    $T_I = T_{I-1} + \Delta T_I$;
    assign $F_I$;                                               Pick their random floor
Elevator-based variables
    Count # of elevators currently in use: those with $T_R[j] < T_I$ for $j = 1 \cdots J$;
    Count # of dwelling elevators: $J_C = M - $(# of those with $T_0[j] + T_d > T_I$);
    if (found a dwelling elevator $j$) and ($P[j] < P_c$)
        then
            $P[j] \to P[j] + 1$;
            $F_{\max}[j] \to \max(F_{\max}[j], F_I)$;
            $T_R[j] \to T_0[j] + 2F_{\max}[j]T_f + P[j]T_s + T_d$;  fi
    if (No dwelling elevators)
        then if (found a returned elevator $j$ with $T_R[j] < T_I$)
            then
                $F_{\max}[j] = F_I$;
                $T_0[j] = T_I$;
                $P[j] = 1$;
                $T_R[j] \to T_0[j] + 2F_{\max}[j]T_f + P[j]T_s + T_d$;  fi
        if (No dwelling elevators) and (No returned elevators)
            then
                $J \to J + 1$;                                  Use a fresh, un-used elevator
                if ($J > M$)                                    OOPS, none were left!
                    then Make the passenger wait for the next returned one
                        $J \to M$;
                        Find the next-returning elevator $j$;
                        $F_{\max}[j] = F_I$;
                        $T_0[j] = T_R[j]$;
                        $P[j] = 1$;
                        $T_R[j] \to T_0[j] + 2F_{\max}[j]T_f + P[j]T_s + T_d$;  fi
            else                                                No waiting, get on the new fresh un-used one
                $F_{\max}[j] = F_I$;
                $T_0[j] = T_I$;
                $P[j] = 1$;
                $T_R[j] \to T_0[j] + 2F_{\max}[j]T_f + P[j]T_s + T_d$;  fi
        fi
Restart from the top of the Main Loop
```