Uncertainty-aware Robustness Assessment of Industrial Elevator Systems

LIPING HAN[*], Nanjing University of Aeronautics and Astronautics, China
SHAUKAT ALI and TAO YUE, Simula Research Laboratory, Norway
AITOR ARRIETA, Mondragon University, Spain
MAITE ARRATIBEL, Orona, Spain

Industrial elevator systems are commonly used software systems in our daily lives, which operate in uncertain environments such as unpredictable passenger traffic, uncertain passenger attributes and behaviors, and hardware delays. Understanding and assessing the robustness of such systems under various uncertainties enable system designers to reason about uncertainties, especially those leading to low system robustness, and consequently improve their designs and implementations in terms of handling uncertainties. To this end, we present a comprehensive empirical study conducted with industrial elevator systems provided by our industrial partner Orona, which focuses on assessing the robustness of a dispatcher, i.e., a software component responsible for elevators' optimal scheduling. In total, we studied 90 industrial dispatchers in our empirical study. Based on the experience gained from the study, we derived an uncertainty-aware robustness assessment method (named *UncerRobua*) comprising a set of guidelines on how to conduct the robustness assessment and a newly proposed ranking algorithm, for supporting the robustness assessment of industrial elevator systems against uncertainties.

CCS Concepts: • **Software and its engineering → Software performance**; **Empirical software validation**; **Software testing and debugging**; • **Computer systems organization → Embedded and cyber-physical systems**.

Additional Key Words and Phrases: Uncertainty-aware Robustness Assessment, Empirical Study

ACM Reference Format:
Liping Han, Shaukat Ali, Tao Yue, Aitor Arrieta, and Maite Arratibel. 2022. Uncertainty-aware Robustness Assessment of Industrial Elevator Systems. *ACM Trans. Softw. Eng. Methodol.* 1, 1 (January 2022), 52 pages.

1 INTRODUCTION

An elevator system is a complex Cyber-Physical System (CPS) responsible to efficiently transport passengers between floors of a building, while ensuring the reliability of its operation and maintaining the passengers' comfort at all time. Such an elevator system is equipped with a dispatcher – a software component for this purpose. Such a dispatcher operates in a constantly changing environment; thus, it is expected to be robust even when facing all kinds of uncertainties such as passengers' uncertain attributes (e.g., *Mass*), and behaviors (e.g., *Loading* and *Unloading Times*)

[*]Also with Simula Research Laboratory.

Authors' addresses: Liping Han, liping@simula.no, Nanjing University of Aeronautics and Astronautics, Nanjing, China; Shaukat Ali, shaukat@simula.no; Tao Yue, tao@simula.no, Simula Research Laboratory, Oslo, Norway; Aitor Arrieta, aarrieta@mondragon.edu, Mondragon University, Mondragon, Spain; Maite Arratibel, marratibel@orona-group.com, Orona, Hernani, Spain.

1049-331X/2022/1-ART $15.00
https://doi.org/

robustly. Moreover, the dispatcher's implementation constantly evolves throughout its entire life cycle due to, e.g., addition of new functionality.

Even though, various software engineering methodologies (e.g., [15, 44]) have been applied to assess, optimize and test the robustness of elevator dispatchers, the state of the art and practice still lack systematic ways of assessing the robustness of industrial elevator systems in terms of their capabilities of dealing with various uncertainties.

For example, in our industrial context, we observed that testing in Software in the Loop (SiL) simulation setup of industrial elevators is performed. However, during such simulation-based testing, fixed values for passengers' attributes, e.g., passengers' *Mass*es being all 75 KG (an average person *Mass* in Europe recommended by the Chartered Institution of Building Services Engineers (CIBSE) Guide D [8]) are used. However, as we all know, in practice, values of the passengers' attributes and also behaviors cannot be fixed, and it is very difficult (if possible) to predict the exact *Mass* and *Loading/Unloading Time* of a passenger, at which floor a passenger will register a call and which floor the passenger's destination will be. All these uncertainties have impact on the Quality of Service (*QoS*) of elevator systems, which also reflects the robustness of their dispatchers against such passengers' uncertainties.

To this end, we first investigated passengers' uncertainties for systematically assessing the robustness of industrial elevators' dispatchers from various aspects with a comprehensive empirical study performed with 90 dispatchers provided by our industrial partner - Orona[1]. Note that we focus exclusively on a dispatcher – an essential software responsible for optimally scheduling elevators, and 89 versions of it. The dispatcher is a real software component from Orona, whereas the rest (e.g., hardware) was simulated with a commercial simulation tool – Elevate[2]. The goal of the empirical study is to understand the robustness of the 90 dispatchers (the original one plus its 89 versions) when facing passengers' uncertainties from the point of view of elevator engineers in the context of SiL simulations with Elevate.

Results of the empirical study indicate that it is very necessary to 1) assess the robustness of different dispatcher versions across various uncertain situations and across multiple traffic templates (e.g., *UpPeak*, which models the traffic flow of morning rush hours in an office building), such that elevator engineers can select the most robust dispatcher version for deployment; 2) investigate which uncertain situation(s) lead to the degradation of the robustness of dispatchers more than others, so as to prioritize optimization scenarios, e.g., designing specific uncertainty handling strategy; and 3) prioritize *QoS* metrics for the purpose of optimizing the robustness of a dispatcher in terms of those *QoS* metrics that are mostly impacted by uncertainties.

Based on these observations from the empirical study, we derive an Uncertainty-aware Robustness assessment method, named *UncerRobua*, which contains a set of guidelines derived based on a comprehensive empirical study and a novel statistical ranking algorithm. We expect *UncerRobua* will be valuable for elevator designers to systematically study the robustness of elevator systems under various uncertain situations and design highly dependable dispatchers.

Contributions. 1) We conducted a comprehensive empirical study with 90 dispatchers from Orona for assessing their robustness against passengers' uncertainties in a systematic and comprehensive manner. The empirical study was enabled with an uncertain traffic profile generator, a robustness quantifier, and a newly-proposed statistical difference-based grouping algorithm (*SD-G*);

2) Based on the results of the empirical study, we derived *UncerRobua* consisting of a set of guidelines for guiding practitioners to assess the robustness of industrial elevator systems against passengers' uncertainties.

[1]https://www.orona-group.com/
[2]https://peters-research.com/index.php/elevate/

Paper Structure. The rest of the article is organized as follows. Section 2 provides background details on the industrial elevator system from Orona, and a running example. Section 3 presents all the details about the design of the empirical study. Results of the empirical study are reported in Section 4. *UncerRobua*'s guidelines and generalization to other domains are presented in Section 5. Sections 6 and 7 present the threats to the validity and related work, respectively. We conclude the article in Section 8.

2 BACKGROUND

In this section, we first discuss the industrial elevator system from Orona (Section 2.1), followed by the presentation of a running example (Section 2.2).

2.1 Industrial Elevator Systems from Orona

With more than 250,000 elevator installations worldwide, Orona[3] is one of the leading elevator manufacturers in Europe. As the majority of CPSs, most of the functionality of a system of elevators is implemented through software. Fig. 1 depicts a simplified version of a system of elevators. In such a system, there are different computing devices, each of them aiming to carry out one or more task. The communication among these devices is done through a Controller Area Network (CAN) bus. At each floor of the building, there is at least one *pushbutton* panel. Conventional elevator systems include *pushbuttons* indicating the direction in which a passenger intends to travel (i.e., to up floors or to down floors), whereas destination control elevator systems include *pushbuttons* with the exact floor at which a passenger intends to travel. In addition, each elevator cabin has a microcontroller in charge of controlling 1) the engines and drives for transporting passengers vertically and 2) the opening and closing of doors. Lastly, there is a component named *Traffic Master* that controls the elevator assignment to each call. A Traffic Master has a *dispatcher* deployed – the key software component for scheduling elevators. The dispatcher receives passengers' calls from the *pushbutton* panel through the CAN bus and assigns an elevator to each call. To make this assignment optimal, the dispatcher uses information of each elevator (e.g., the estimated number of passengers inside the elevator, the direction of the elevator, doors' status). Note that, in this paper, we focus exclusively on assessing the robustness of dispatchers, i.e., software.

Orona has an extensive suite of dispatchers to provide solutions to the different demands of their customers. Moreover, the software constantly evolves to deal with implementing new functionalities, hardware obsolescence, correction of bugs, etc. The current testing process of the different dispatchers' versions contemplates three main levels. The first level refers to the SiL test level, for which a tool named Elevate is used for simulation-based testing. In this case, both the software and the hardware components of the elevators are simulated on a personal computer. The second level refers to the Hardware in the Loop (HiL) test level. At this level, on the one hand, the software is integrated into the real-time target processor, along with all the real-time infrastructure (e.g., real-time operating system, drivers). On the other hand, most of the hardware is the one used in the real installation. In addition, some mechanical and electrical parts (e.g., engines, drives) are emulated through real-time test benches. The last step is the validation in operation. When the new software version is considered to be ready (i.e., thoroughly tested at the SiL and HiL test levels), a software maintainer goes to the installation and carries out a set of manual tests.

In this paper, we focus at the SiL test level. As mentioned above, tests at this level are executed through the domain-specific simulator Elevate. A test case in the context of Orona consists of 1) building installation and 2) passenger file. The former refers to the characteristics of a building, and includes information like the number of floors, number of elevators, capacity of each of the

[3]https://www.orona-group.com/

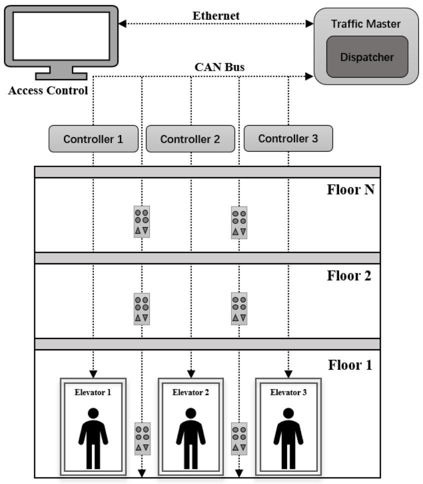


Fig. 1. A simplified elevator system

elevators, maximum speed of each elevator, etc. The latter refers to a list of passengers in a building traveling from one floor to another. Each passenger has a set of attributes (explained in Section 2.2), which includes the time at which they arrive at the floor, their *Mass*, *Loading* and *Unloading Time*s, etc. Some of the attributes in the passenger file of a test case are fixed as recommended by different elevator industry guidelines (e.g., CIBSE Guide D[8]). For instance, the *Mass* of all passengers is set to 75 KG, and the *Loading* and *Unloading Time*s to 1.2 seconds. However, in practice, such attributes are uncertain to the dispatcher (e.g., the dispatcher does not know the *Mass* of the passenger behind a call). In this paper, we conjecture that such uncertain attributes may have an influence on the **robustness** (see Definition 1) of the elevator systems.

Definition 1. ***Robustness****, in our context, refers to the degree to which a dispatcher's functional performance is not affected in the presence of passengers' uncertainties. The functional performance of elevator systems is measured with the quality of service provided to an individual passenger (see Definition 2) and all passengers together (see Definition 4).*

In this domain, *QoS* metrics are defined based on how passengers in an installation perceive whether elevators perform well. For instance, studies have shown that psychologically, the time that passengers spend waiting for an elevator is a critical aspect for them in the perception of the quality of provided services [24]. Generally, most elevator systems are dedicated to optimize various times while providing services so as to improve passenger's satisfaction. The time that determines passenger's satisfaction are: Waiting Time (*WT*), Transit Time (*TT*) and Time to Destination (*TD*), which are metrics of *QoS* provided to a single passenger, as defined in Definition 2. Based on all passengers' *WT*, *TT* and *TD* in a certain period of time (*Time List*, see Definition 3), various metrics can be calculated, i.e., *AWT*, *LWT*, *ATT*, *LTT*, *ATD* and *LTD* (see Definition 4) [8].

Definition 2. ***Metrics of QoS provided to an individual passenger (*** QoS_{indivP} ***)***

- ***Waiting Time** (**WT**) is the time between the passenger registering a call on a certain floor until the elevator arrives and its door begins to open. If the elevator door is already opened when the passenger arrives, then the WT of this passenger is 0.*
- ***Transit Time** (**TT**) is the time between the elevator door begins to open until the elevator reaches the passenger's destination floor and begins to open elevator door again.*

- ***Time to Destination (TD)*** *is the sum of passenger's waiting time and transit time.*

Definition 3. ***Time List*** *is a sequence of all passengers' WT, TT or TD within a certain period of time, represented as* $TL = \{TL_s | s = 1, 2, 3\}$, *where* TL_s *is the* s*th Time List. More specifically,* $TL_1 = \{wt_c | c = 1, ..., NP\}$, $TL_2 = \{tt_c | c = 1, ..., NP\}$ *and* $TL_3 = \{td_c | c = 1, ..., NP\}$, *where,* wt_c, tt_c *and* td_c *are the waiting time, transit time and time to destination of the* c*th passenger,* NP *is the number of total passengers requested services within the time period.*

Definition 4. ***Metrics of QoS derived from all passengers together (*** QoS_{allP} ***)***

- ***AWT*** *is the average waiting time of all passengers whose calls have been answered within a certain period of time, represented as* $AVERAGE(TL_1)$.
- ***LWT*** *is the longest waiting time experienced by a passenger within a certain period of time, represented as* $MAX(TL_1)$.
- ***ATT*** *is the average transit time of all passengers who completed their journeys within a certain period of time, represented as* $AVERAGE(TL_2)$.
- ***LTT*** *is the longest transit time experienced by a passenger who completed her/his journey within a certain period of time, represented as* $MAX(TL_2)$.
- ***ATD*** *is the average time to destination of all passengers within a certain period of time, represented as* $AVERAGE(TL_3)$.
- ***LTD*** *is the longest time to destination experienced by a passenger within a certain period of time, represented as* $MAX(TL_3)$.

2.2 Running Example

Table 1 presents an excerpt of the *UpPeak*[4] traffic profile with ten passengers provided by Elevate, which we use as the running example to illustrate various concepts and calculations in the rest of the paper. Note that passenger traffic values (i.e., values for *Arrival Time*, *Arrival Floor* and *Destination Floor*) are generated based on a selected traffic template (e.g., *UpPeak*), while values for *Mass*, *Capacity Factor*, *Loading Time* and *Unloading Time*) are consistently configured the same for all the passengers. From the table, one can see that most passengers with different destinations arrive on the first floor, e.g., in the morning during a weekday in an office building. For example, the arrival time of the first passenger is 30691, also denoted as 08:31:31. The conversion is: 8×60×60+31×60+31.

Once the simulation is completed, Elevate outputs simulation reports. Table 2 and Table 3 show key results corresponding to the *UpPeak* traffic profile in Table 1. Table 2 presents details of each passenger's journey in the simulation reports. Results include: (1) which elevator responded to the call (e.g., the 2nd elevator responded to the first passenger's call, as shown in the first row of column *Elevator Used*); (2) at which time an elevator arrived or reached a destination (shown in columns *Time Elevator Arrived* and *Time Reached Destination*). For instance, for the first passenger, the 2nd elevator arrived at 8:31:31 and the passenger reached the destination floor at 8:31:45. In addition, Table 2 also provides derived time information: *WT* and *TT*. For instance, for the first passenger, the *WT* and *TT* are 0s and 14.1s, respectively. Moreover, in Table 3, we present results of all the six QoS_{allP} metrics. Note that *TD* can be derived from *WT* and *TT* (see Definition 3) and consequently all the QoS_{allP} metrics can be calculated with the formulas in Definition 4.

3 DATASET PREPARATION AND EXPERIMENT DESIGN

In this Section, we present, in detail, the dataset preparation and the experiment design and execution of the empirical study. Fig. 2 shows the overall process of the empirical study. In the rest of this section, we first introduce how we prepared the dataset used for the empirical study

[4]https://elevate.helpdocsonline.com/peters-research-cibse-modern-office-up-peak

Table 1. Running Example: Passenger data of ten passengers during *UpPeak*. *Mass* is in KG; *Capacity Factor* is in percentage; *Loading Time* and *Unloading Time* are in seconds. Column *ID* distinguishes different passengers, which is added for better illustration and it does not exist in files produced by Elevate.

ID	Arrival Time	Arrival Floor	Destination Floor	Mass	Capacity Factor	Loading Time	Unloading Time
Pa1	30691 (8:31:31)	1	2	75	70	1.2	1.2
Pa2	30703 (8:31:43)	6	1	75	70	1.2	1.2
Pa3	30705 (8:31:45)	1	13	75	70	1.2	1.2
Pa4	30712 (8:31:52)	1	3	75	70	1.2	1.2
Pa5	30727 (8:32:07)	1	12	75	70	1.2	1.2
Pa6	30727 (8:32:07)	1	2	75	70	1.2	1.2
Pa7	30735 (8:32:15)	1	6	75	70	1.2	1.2
Pa8	30737 (8:32:17)	1	14	75	70	1.2	1.2
Pa9	30738 (8:32:18)	1	5	75	70	1.2	1.2
Pa10	30742 (8:32:22)	2	1	75	70	1.2	1.2

Table 2. Running Example: Partial simulation results corresponding to Table 1. *WT* and *TT* are in seconds. The *ID* column is added to distinguish different passengers.*

ID	Passenger Attributes	Elevator Used	Time Elevator Arrived	Time Reached Destination	WT	TT
Pa1	–	2	8:31:31	8:31:45	0	14.1
Pa2	–	3	8:31:55	8:32:16	12.2	20.4
Pa3	–	4	8:31:45	8:32:36	0.1	50.9
Pa4	–	4	8:31:52	8:32:08	0.1	15.9
Pa5	–	5	8:32:07	8:33:11	0.1	64.2
Pa6	–	5	8:32:07	8:32:31	0.1	23.4
Pa7	–	3	8:32:15	8:32:49	0.1	34.3
Pa8	–	3	8:32:15	8:33:27	0	70.2
Pa9	–	3	8:32:16	8:32:36	0	17.8
Pa10	–	2	8:32:22	8:32:36	0.1	14.1

**Passenger Attributes* of each passenger (uniquely identified with its *ID*) are provided in Table 1.

Table 3. Running Example: values of QoS_{allP} metrics corresponding to Table 1 and Table 2. All the QoS_{allP} values are in seconds.

AWT	LWT	ATT	LTT	ATD	LTD
1.3	12.2	32.5	70.2	33.8	70.2

(Section 3.1). Then we discuss our research questions (Section 3.2), followed by the evaluation metrics (Section 3.3). Next, we present the *SD-G* test (Section 3.4), which we used, as part of the evaluation metrics, for addressing the research questions. Finally, we describe the parameter settings and how we conducted the empirical study (Section 3.5).

3.1 Dataset Preparation

This paper focuses on uncertainties related to passengers while analyzing the robustness of dispatchers in the SiL simulation from various aspects. To explain how we use the SiL simulation to

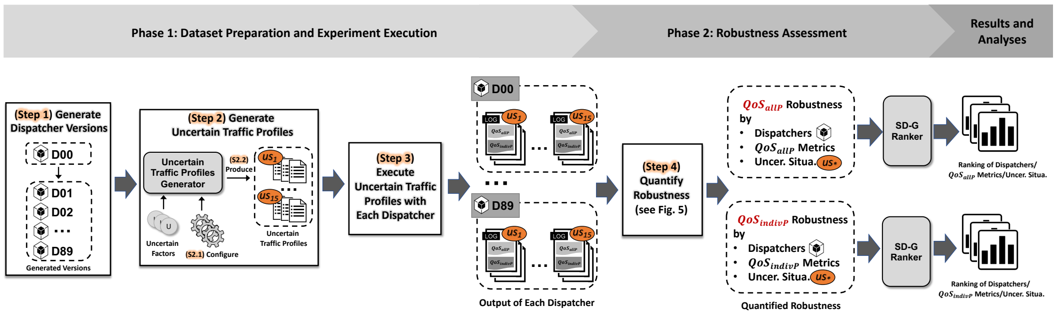

Phase 1: Dataset Preparation and Experiment Execution
Phase 2: Robustness Assessment
Results and Analyses
(Step 1) Generate Dispatcher Versions
D00
D01
D02
D89
Generated Versions
(Step 2) Generate Uncertain Traffic Profiles
Uncertain Traffic Profiles Generator
(S2.2) Produce
Uncertain Factors
(S2.1) Configure
Uncertain Traffic Profiles
(Step 3) Execute Uncertain Traffic Profiles with Each Dispatcher
LOG
Output of Each Dispatcher
(Step 4) Quantify Robustness (see Fig. 5)
QoS_{allP} Robustness by
Dispatchers
QoS_{allP} Metrics
Uncer. Situa.
QoS_{indivP} Robustness by
QoS_{indivP} Metrics
Quantified Robustness
SD-G Ranker
Ranking of Dispatchers/ QoS_{allP} Metrics/Uncer. Situa.
Ranking of Dispatchers/ QoS_{indivP} Metrics/Uncer. Situa.

Fig. 2. Overview of the Empirical Study. Note that us_i represents the ith ($i = 1, .., 15$) uncertain situation; Steps 1-4 are presented in Sections 3.1.2, 3.1.3, 3.5, and 3.3, respectively.

generate the dataset required for the robustness analysis, first, we provide definitions of the key terminologies in Section 3.1.1, followed by the generation of the 89 versions of the dispatcher in Section 3.1.2. We then present our algorithm for the generation of the uncertain traffic profiles in Section 3.1.3, and present the settings for the dataset generation in Section 3.1.4. Finally we describe how we perform the generation with simulations in Section 3.1.5.

3.1.1 *Terminology.*

Definition 5 (**Uncertain Factor**). *All the attributes in Table 1 that define a traffic profile are uncertain factors (Section 2.2). For example, it is uncertain when a passenger arrives at a specific floor; thereby Arrival Time is an uncertain factor.*

Definition 6 (**Uncertain Traffic Profile**). *An uncertain traffic profile is a traffic profile with uncertainty, i.e., uncertain values of one or more uncertain factor(s). For example, an update to column Mass in Table 1 with uncertain values can generate an uncertain traffic profile caused by single uncertain factor Mass.*

Definition 7 (**Uncertain Situation**). *An uncertain situation represents uncertainties due to a single uncertain factor or an interaction among two or more uncertain factors. Assuming an uncertain factor set U with un ($un > 1$) number of uncertain factors, all the* ***t-way interactions*** *(i.e., $t = \{1, 2, \dots un\}$) among all the uncertain factors in U means $\sum_{t=1}^{un} C_{un}^{t}$ number of total uncertain situations. We use, in the rest of the paper,* ***us**** *to represent an uncertain situation, $*$ is the abbreviation(s) of the uncertain factor(s).*

*3.1.2 Generation of the Dispatchers (**Step 1** in Fig. 2).* We employed 90 dispatchers (one original dispatcher and its 89 generated versions). The 89 versions were manually generated by a domain expert based on the original program with a small syntactic variation, such as a relational operator change (e.g., "<" changed by ">=") or an arithmetic operator change (e.g., "+" by "-"). The dispatcher was implemented in C, therefore, classical mutation operators for C language [1] were used. The domain expert initially generated 99 versions in a uniform manner throughout relevant sections of the source code for the simulation environment. All dispatcher versions were reviewed by another domain expert to ensure that the generated versions were not semantically equivalent to the original program. Ten out of the 99 versions led the test execution not to finish, e.g., due to not assigning certain calls, thus are faulty versions. Therefore, these 10 versions were removed, having a final set of 89 versions, which are executable, to be employed for our experiments. Note that these 89 versions have no functional failures, e.g., leaving a call unattended, but rather resulting in possibly sub-optimal elevator assignments affecting *QoS*. In other words, these generated versions can return an assignment for each call that may not provide optimal *QoS*. In the rest of the paper, we use *D01-D89* to represent the 89 generated versions of the original dispatcher (*D00*).

Note that the implementation of a dispatcher is deterministic, i.e., given the same input and configuration, the dispatcher always outputs the same elevator assignment. Thus, there is no inherent uncertainty in the dispatcher. On the other hand, the same inputs and configurations given to different dispatchers would likely produce different elevator assignments of different *QoS*.

*3.1.3 Generation of Uncertain Traffic Profiles (**Step 2** in Fig. 2).* As shown in Fig. 2, we need to generate uncertain traffic profiles based on captured (or interested) uncertain factors with *Uncertain Traffic Profiles Generator*. Taking an example of uncertain factor *Mass*, it takes a value within an interval with the *min* and *max* values, e.g., being 60 KG and 90 KG, respectively. Other uncertain factors are also captured as intervals. We implemented a mechanism by setting an interval around the recommended average values of these factors for a certain region (e.g., Europe). For instance, a widely used CIBSE Guide D [8] recommends to use 75 KG as the average *Mass* for the passengers for the simulation of elevators in Europe.

Algorithm 1 presents the step-wise details for generating uncertain traffic profiles. Assuming an uncertain factor set $U = \{u_i | i = 1, 2, ..., un\}$, where u_i is the ith uncertain factor (e.g., uncertain factor *Mass*, see Definition 5), and un is the total number of uncertain factors. Correspondingly, all uncertain situations caused by single uncertain factor and t-way interactions (see Definition 7) among several uncertain factors in uncertain factor set U can be represented by set $US = \{us_o | o = 1, 2, ..., NS\}$, where us_o is the oth uncertain situation caused by U, NS is the number of uncertain situations, there are $NS = \sum_{t=1}^{un} C_{un}^{t} = 2^{un} - 1$ uncertain situations in total, which can be simulated by $2^{un} - 1$ types of uncertain traffic profiles (see Definition 6). For example, to investigate uncertain factor set U with three uncertain factors $\{u_1 = \textit{Mass}\ (M), u_2 = \textit{Loading Time}\ (L), u_3 = \textit{Unloading Time}\ (U)\}$, the total number of uncertain situations caused by U is 7, of which 3 (C_3^1) are caused by single uncertain factor $\{us_1 = usM, us_2 = usL, us_3 = usU\}$, 3 ($C_3^2$) are caused by 2-way interactions $\{us_4 = usM - L, us_5 = usM - U, us_6 = usL - U\}$ and 1 is (C_3^3) caused by 3-way interactions $\{us_7 = usM - L - U\}$. Studying interactions among uncertain factors is important since effects (e.g., degraded *QoS*) might only be visible when a dispatcher faces uncertainties from a less number of uncertain factors simultaneously. This is because such effects might be masked by certain uncertain factors involved in some uncertain situations. Thus, we study and provide evidence of how robustness is affected by various interactions among uncertain factors in this empirical study.

We generate uncertain traffic profiles of an uncertain situation by generating values of its uncertain factors corresponding to their respective predefined intervals around the recommended values in CIBSE Guide D [8] (e.g., the recommended value for *Mass* in Europe is 75 KG). With the recommended values, a set of intervals for the uncertain factors are defined, represented as $I = \{I_i | i = 1, 2, ..., un\}$, where, $I_i = [a_i, b_i]$ is the interval for uncertain factor u_i. The minimum and maximum values of the interval are chosen by $\pm x$ based on the recommended value. For example, an interval $[60, 90]$ for uncertain factor *Mass* can be obtained by choosing ± 15 based on the recommended value of *Mass* (75 KG).

For a set of uncertain factors U, all the generated uncertain traffic profiles can be represented as $P = \{P_{us_o} | o = 1, ..., NS\}$, where P_{us_o} is the set of generated uncertain traffic profiles corresponding to uncertain situation us_o. For each uncertain situation, we repeat the generation NR times. This means that each uncertain situation us_o has NR uncertain traffic profiles, represented as $P_{us_o} = \{P_{us_o,k} | o = 1, ..., NS; k = 1, ..., NR\}$, where $P_{us_o,k}$ is the kth uncertain traffic profile corresponding to the oth uncertain situation us_o. Each uncertain traffic profile contains NP number of passengers

Algorithm 1: Uncertain Traffic Profiles Generation

```
Input: Uncertain factors $U$, Intervals $I$, number of repetitions per generation $NR$, baseline profile $P_b$
Output: Uncertain traffic profiles $P$
for $us_o$ in $US$ do
    for $k \leftarrow 1$ to $NR$ do
        $P_{us_o,k} \leftarrow P_b$ // Initialize the uncertain traffic profile with the baseline profile
        $SatisfyConstraint \leftarrow$ FALSE // Initialize the variable marking the constraint state
        while $!SatisfyConstraint$ do
            for $Pa_{us_o,k,c}$ in $P_{us_o,k}$ do
                for $u_i$ in $us_o$ do
                    generate value of $u_i$ for $Pa_{us_o,k,c}$ using Eq. 1
            $Ave_{u_i} \leftarrow \frac{1}{NP}\sum_{c=1}^{NP} v_{i,c}$ // Calculate the average value of the $i$th uncertain factor
             ($u_i$) of all the passengers
            if $Ave_{u_i}$ of $P_{us_o,k}$ equals to the value of $u_i$ in $P_b$ then // Check if the generated profile
             satisfies the constraint on the average
                save $P_{us_o,k}$
                $SatisfyConstraint \leftarrow$ TRUE
```

(see example in Table 1), represented as $P_{us_o,k} = \{Pa_{us_o,k,c} | o = 1, ..., NS; k = 1, ..., NR; c = 1, ..., NP\}$, where $Pa_{us_o,k,c}$ is the cth passenger in uncertain traffic profile $P_{us_o,k}$.

As shown in Lines 2-8 in Algorithm 1, to generate an uncertain traffic profile, for each passenger, we generate a value for each uncertain factor forming the uncertain situation (e.g., the columns in Table 1). Formally, the value of uncertain factor u_i of the cth passenger $Pa_{us_o,k,c}$ in uncertain traffic profile $P_{us_o,k}$ can be generated within its predefined interval I_i, represented as Eq. 1.

$$v_{i,c} = \underset{a_i \le x \le b_i}{Generate}(x) \tag{1}$$

To ensure that generated uncertain traffic profiles do not deviate too much from the baseline profile which has fixed values of the uncertain factors for each passenger, we check the average of all the generated values for each uncertain factor, and keep the generated profile having the same average value of each uncertain factor as the baseline profile. Otherwise, we iterate the generation process until obtaining a matching profile (Lines 5-12 in Algorithm 1). Taking the example profile in Table 1 for instance, if we want to generate an uncertain traffic profile corresponding to uncertain situation *usM*, we need to generate 10 *Mass* values for the 10 passengers and check whether the generated profile has the average value of the 10 *Mass* values equal to 75 KG. If not, we iterate the generation process until we obtain a profile that satisfies the constraint on the average value.

3.1.4 *Dataset Generation Settings (**S2.1** in Fig. 2).*

We used an office building configuration from Section 4.8 of the CIBSE Guide D [8], which contains 6 elevators, 14 floors, and 1120 persons (80 persons per floor). Each elevator has settings as shown in Table 4.

Table 4. Elevator Configuration

Capacity (KG)	Car area (m^2)	Speed (m/s)	Acceleration (m/s^2)	Jerk (m/s^3)
1600	3.56	2.5	0.8	1.0

We used two traffic templates: modern office lunch peak[5] (*LunchPeak*) and modern office up peak[6] (*UpPeak*) to generate two baseline traffic profiles. *LunchPeak* models the passenger traffic during lunch peak. More specifically, it models the traffic flow between 12:15 and 13:15 with nearly half incoming, half outgoing and minimum inter-floors traffic, comprising 1409 passengers. Different from *LunchPeak*, *UpPeak* models the traffic flow between 08:30 and 09:30 with a majority of incoming traffic, comprising 1150 passengers. Passenger activities of *LunchPeak* and *UpPeak* generated by Elevate are shown in Fig. 3 and Fig. 4, respectively. Obviously, the traffic flows during *LunchPeak* and *UpPeak* are completely different. That's because most people came to the offices for work around 9:00am, went out for lunch, then came back after lunch during the lunch break.

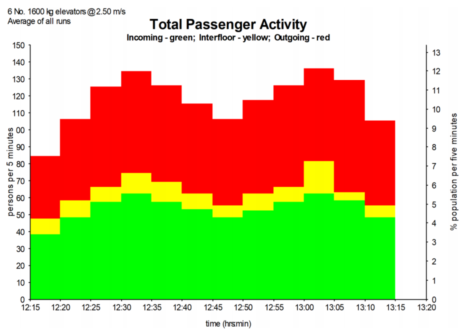


Fig. 3. *LunchPeak* Passenger Activity.

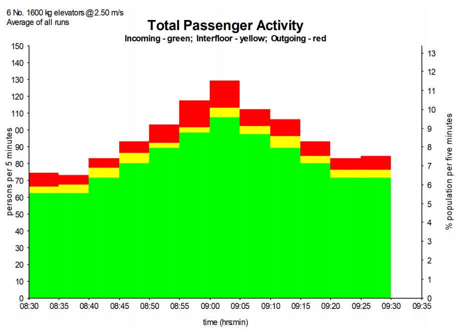


Fig. 4. *UpPeak* Passenger Activity.

In addition to configuring building and elevators, and selecting traffic templates, passenger details are also needed for simulations. We set *Mass* to 75 KG (i.e., the average *Mass* in Europe [8]), *Loading* and *Unloading Times* to 1.2 seconds (i.e., the average time to enter and exit an elevator according to [8]). Given that the maximum *Capacity Factor* is usually 80% [8], we set the *Capacity Factor* to 70% based on examples from [8]. After completing all the settings, we ran Elevate and obtained the **baseline profiles** (for the *LunchPeak* and *UpPeak*), which were used to generate uncertain traffic profiles. Note that a set of values for passengers' *Arrival Time*, *Arrival Floor* and *Destination Floor* are generated by Elevate based on a selected traffic template, which models passengers' activities in reality based on multiple surveys of operational elevators in buildings conducted by Peters Research Ltd [7]. Therefore, the distribution of traffic follows a real-world distribution. Consequently, in our study, we keep their default values in Elevate for all the simulations. Therefore, in our study, these three factors are not considered as independent variables. Instead, *Mass* (M), *Capacity Factor* (C), *Loading Time* (L) and *Unloading Time* (U) of each passenger in the generated traffic profile are kept the same by Elevate. This is also the current practice at Orona. Thus, in our study, we consider these four parameters as the uncertain factors and varied their values to study their impact on the robustness of dispatchers. Note that these are the only four uncertain factors related to passengers that can be controlled in Elevate and are modeled as numerical values. We also like to emphasize that the real distributions of *Mass*, *Capacity Factor*, *Loading Time* and *Unloading Time* do not exist in our industrial context since elevators do not record these for individual passengers.

*3.1.5 Dataset Generation (**S2.2** in Fig. 2).* With the baseline profiles, we generate uncertain traffic profiles based on the studied uncertain factors. We generate values of uncertain factors *Mass*,

[5]https://elevate.helpdocsonline.com/peters-research-cibse-modern-office-lunch-peak
[6]https://elevate.helpdocsonline.com/peters-research-cibse-modern-office-up-peak
[7]https://peters-research.com

Loading Time and *Unloading Time*, based on their intervals (Table 5). These intervals are obtained by setting their lower and upper bounds as -20% and +20% of their baseline values recommended in CIBSE Guide D [8]. For instance, the lower and upper bounds of *Mass*'s interval is $75 - 20\% \times 75 = 60$ and $75 + 20\% \times 75 = 90$, respectively. For *Capacity Factor*, its maximum value is usually 80% [8] and the value of the baseline is 70%. To avoid generated values exceeding 80%, we use ±10 to define its interval. All these intervals have been discussed and confirmed with Orona, to ensure that they are realistic. Consequently, the four uncertain factors lead to $NS = 15$ uncertain situations (i.e., the size of US being **15**), which are all the uncertainty cases being considered in our study, including 4 single-way (t=1) and 11 multiple-way (t>1) interactions. For each uncertain situation, we generated $NR = 10$ traffic profiles. Correspondingly, each type of traffic template (i.e., *LunchPeak* and *UpPeak*) has 150 uncertain traffic profiles. We obtained 300 (150×2) uncertain traffic profiles in total.

Note that, in practice, elevator engineers use fixed values from the standards, e.g., CIBSE Guide D [8], to set passengers' attributes in Elevate. Moreover, there are not any recommendations on distributions of these attributes available with which the four uncertain factors should conform to. More specifically, for *Mass*, existing studies on human *Mass* distributions are subject to various conditions such as being specific to occupation [23], age, and gender [25, 26]. Passengers' *Loading Time* and *Unloading Time* have been studied in two real scenarios in [16], but their distributions were not investigated. Regarding *Capacity Factor*, the related works (e.g., [35, 48, 58]) used fixed values.

Table 5. Intervals of Uncertain Factors

Uncertain Factors	Baseline	Interval
Mass (KG)	75	[60, 90]
Capacity Factor (%)	70	[60, 80]
Loading Time (s)	1.2	[0.96, 1.44]
Unloading Time (s)	1.2	[0.96, 1.44]

3.2 Goal and Research Questions

We employed the Goal-Question-Metric (GQM) method proposed by Wholin et al. [52] to describe goals, research questions, and metrics for our empirical study.

Goal. The purpose of this study is to understand the robustness of the dispatchers in the presence of passengers' uncertainties from the point of view of elevator engineers in the context of SiL with industrial dispatchers deployed. Based on this goal, we aim to answer the following **research questions** (RQs).

- **RQ1: How robust are dispatchers under passengers' uncertainties when measured with *QoS* based on the quality of service provided to all the passengers together?** Note that in the rest of the paper, we call this type of robustness as QoS_{allP} robustness to be concise. This RQ is further answered with the following three sub-RQs:
  - **RQ1.1: How does the QoS_{allP} robustness of the dispatchers vary when dealing with passengers' uncertainties?** This RQ aims to provide elevator engineers an overview of the assessment of each dispatcher's robustness, such that they are more informed when choosing dispatcher(s) with the best QoS_{allP} robustness for targeted application contexts.
  - **RQ1.2: Do the dispatchers perform differently under the various uncertain situations in terms of the QoS_{allP} robustness?** With this RQ, we want to investigate whether the different uncertain situations have different degrees of impact on the QoS_{allP} robustness

of the dispatchers. If so, which uncertain situation has the most or the least impact. In addition, we study the correlation of the number of uncertain factors in the uncertain situations with the QoS_{allP} robustness observed. Overall, with this RQ, we aim to provide elevator engineers information about under which uncertain situations, the dispatchers might exhibit a degraded QoS_{allP} robustness.

  - **RQ1.3: Is the robustness of the dispatchers different when measured with different QoS_{allP} metrics, when dealing with passengers' uncertainties?** With this RQ, we aim to investigate to which degree each QoS_{allP} metric is impacted by passengers' uncertainties by providing a ranking of the QoS_{allP} metrics. This RQ is important for elevator engineers to know which QoS_{allP} metric(s) should be targeted when optimizing dispatchers.

- **RQ2: How robust are the dispatchers under passengers' uncertainties when measured with QoS based on the quality of service provided to each passenger individually?** We call this type of robustness as the QoS_{indivP} robustness in the rest of the paper. Note that RQ1 studies the robustness of the dispatchers with QoS_{allP} (which considers the quality of services provided to all the passengers as a whole), whereas RQ2 studies the QoS_{indivP} robustness (from the perspective of each individual passenger). This RQ is further answered with the following three sub-RQs below:
  - **RQ2.1: How does the QoS_{indivP} robustness of the dispatchers vary in terms of handling passengers' uncertainties?** This RQ aims to provide elevator engineers an overview of the assessment of their dispatchers' QoS_{indivP} robustness such that they are more informed when choosing dispatcher(s) with the best QoS_{indivP} robustness for targeted application contexts.
  - **RQ2.2: Do the dispatchers perform differently under the various uncertain situations with respect to the QoS_{indivP} robustness?** With this RQ, we first want to investigate whether the different uncertain situations have different degrees of impact on the QoS_{indivP} robustness of the dispatchers. If so, we further investigate which uncertain situation has the most or least impact on the QoS_{indivP} robustness. Furthermore, we study if there is a correlation between the number of the uncertain factors involved in the uncertain situations and the QoS_{indivP} robustness of the dispatchers. Same as for RQ1.2, with this RQ, we aim to provide elevator engineers information about under which uncertain situations, the dispatchers have a higher chance of showing a degraded QoS_{indivP} robustness.
  - **RQ2.3: Is the robustness of the dispatchers different when measured with the different QoS_{indivP} metrics?** Here, we aim to investigate which QoS_{indivP} metric is impacted the most or least. Same as for RQ1.3, answering this RQ provides elevator engineers insights about which QoS_{indivP} metric(s) should be the next optimization target.

3.3 Evaluation Metrics

We define a set of evaluation metrics for answering RQ1 and RQ2 in Section 3.3.1 and Section 3.3.2, respectively. The mapping of the RQs and the metrics is summarized in Table 6.

3.3.1 *Evaluation Metrics for RQ1.*

Recall from Section 3.2 that RQ1 studies the robustness based on the QoS_{allP} metrics, i.e., *AWT*, *LWT*, *ATT*, *LTT*, *ATD*, and *LTD*. Each dispatcher is evaluated by comparing its robustness under each of the 15 uncertain situations (corresponding to a generated traffic profile with uncertainties introduced on the involved uncertain factors together forming the particular uncertain situation) with the baseline traffic profile of no uncertainty (see Section 3.1.4), in terms of the QoS_{allP} metrics.

First, based on Algorithm 1, we generate a certain number of uncertain traffic profiles, i.e., *NR* (10 in our experiments) for each uncertain situation and simulate them and the baseline profile with fixed values for all the uncertain factors (see Section 3.1.5). For each uncertain situation, we got *NR* values corresponding to each QoS_{allP} metric (e.g., 10 *AWT* values). For the baseline profile, one value corresponding to each QoS_{allP} metric (e.g., 1 *AWT* value) is obtained. Next, we compare the *NR* values corresponding to a QoS_{allP} metric with its respective one value with the one-sample Wilcoxon signed rank test, e.g., 10 *AWT* values corresponding to 10 repetitions of an uncertain situation are compared with one *AWT* value produced by the baseline traffic profile with no passengers uncertainties. We chose the one-sample Wilcoxon signed rank test (see Eq. 3) since our data meet all the prerequisites of the test. We chose the typical significance level of 0.05. A resulting p-value less than 0.05 indicates that the uncertain situation significantly worsens the QoS_{allP} metric, though some of the QoS_{allP} values may still be acceptable to passengers. The significant degradation of the robustness of the dispatcher under uncertainties implies the low QoS_{allP} robustness.

For each sub-research question of RQ1, we assess robustness with two types of metrics from three different perspectives. First, we count the number of cases that p-values are less than 0.05 to study robustness by: 1) considering both the uncertain situations and QoS_{allP} metrics together for each dispatcher (RQ1.1), 2) each uncertain situation (RQ1.2), 3) each QoS_{allP} metric. Note that we count the number of cases that p-values are less than 0.05 to make conclusions at a high-level and do not apply further statistics on them to compare different dispatchers, uncertain situations, or QoS_{allP} metrics. A dispatcher with a higher number of cases where p-values are less than 0.05 indicates a lower robustness, and the counts are comparable across all the dispatchers. The quantification of the QoS_{allP} robustness is visualized in the upper row in Fig. 5. Second, we perform three types of robustness rankings with the *SD-G* test, i.e., ranking of dispatchers (see Definition 8 and Section 4.1.1), uncertain situations (see Definition 9 and Section 4.1.2), and QoS_{allP} metrics (see Definition 10 and Section 4.1.3). The detailed calculations of the metrics are provided in Appendix A for reference. The *SD-G* test is described in Section 3.4.

3.3.2 Evaluation Metrics for RQ2. As described in Section 3.2, RQ2 focuses on studying dispatcher robustness with respect to the QoS_{indivP} metrics (i.e., *WT*, *TT*, and *TD*, as defined in Section 2.1). Different from RQ1, in RQ2, we assess the robustness of a dispatcher in a more fine-grained manner, i.e., from the perspective of each passenger in terms of the quality of services she/he has received, when the dispatcher operates under situations of with and without uncertainties.

We used the simulation results, i.e., a set of values of the QoS_{indivP} metrics for each passenger (see an example in Table 2), generated by the same dataset (see Section 3.1.5) as for RQ1. For each QoS_{indivP} metric, we obtained *NR Time Lists*[8] (i.e., *NR* samples and 10 in our experiments, see Definition 3) corresponding to *NR* repetitions for each uncertain situation and one sample file corresponding to the baseline profile with fixed values for passengers' uncertain factors. Next, we compare each sample corresponding to an uncertain situation with the sample of baseline profile using a paired Wilcoxon signed rank test (see Eq. 4) at the significance level of 0.05. As a result, we obtained NR p-values (i.e., 10 in our experiments), based on which the three sub-research questions are answered, as described below.

First, we count the number of cases that p-values are less than 0.05 to quantify the robustness of each dispatcher (RQ2.1, see Eq. 6 and Eq. 17), by an uncertain situation (RQ2.2, see Eq. 20 and Eq. 21), and by a QoS_{indivP} metric (RQ2.3, see Eq. 25). The calculations are visualized in the row below in Fig. 5. In principle, higher the number of cases that p-values are less than 0.05, higher the likelihood that a dispatcher is less robust. Second, we perform robustness rankings (with the

[8]Note that Elevate only generates *WT* and *TT*, from which *TD* is derived.

SD-G test) of the dispatchers (RQ2.1, see Section 4.2.1), uncertain situations (RQ2.2, see Section 4.2.2), and the QoS_{indivP} metrics (RQ2.3, see Section 4.2.3). Detailed calculations of the metrics are provided in the Appendix A, whereas the employed *SD-G* test is described in Section 3.4.

Table 6. Mapping of the Evaluation Metrics to RQs

Goal: Understanding the robustness of dispatchers in the presence of passengers' uncertainties					
RQ #		**QoS Metric**	**Comparison Baseline (Statistical Test)**	**Evaluation Metrics (Definition #)**	**Measurements (Equation #)**
1	1.1	QoS_{allP} metrics –*AWT, LWT, ATT, LTT, ATD, LTD*	Comparing QoS_{allP} of a dispatcher with and without uncertain traffic profiles (One-Sample Wilcoxon Signed-rank test)	QoS_{allP} robustness for each dispatcher (8)	Count of p-value< 0.05 (5, 7)
				Ranking of dispatchers (8)	*SD-G* test for dispatchers (8, 9)
	1.2			QoS_{allP} robustness of all the dispatchers under each uncertain situation (9)	Count of p-value< 0.05 (10, 11)
				Ranking of uncertain situations (9)	*SD-G* test for uncertain situations (12, 13)
	1.3			All dispatchers' robustness in terms of each QoS_{allP} metric (10)	Count of p-value< 0.05 (15)
				Ranking of QoS_{allP} metrics (10)	*SD-G* test for QoS_{allP} metrics (16)
2	2.1	QoS_{indivP} metrics – *WT, TT, TD*	Comparing QoS_{indivP} of a dispatcher with and without uncertain traffic profiles (Paired Wilcoxon Signed-rank test)	QoS_{indivP} robustness for each dispatcher (11)	Count of p-value< 0.05 (6, 17)
				Ranking of dispatchers (11)	*SD-G* test for dispatchers (18, 19)
	2.2			QoS_{indivP} robustness of all the dispatchers under each uncertain situation (12)	Count of p-value< 0.05 (20, 21)
				Ranking of uncertain situations (12)	*SD-G* test for uncertain situations (22, 23)
	2.3			All dispatchers' robustness in terms of each QoS_{indivP} metric (13)	Count of p-value< 0.05 (25)
				Ranking of QoS_{indivP} metrics (13)	*SD-G* test for QoS_{indivP} metrics (26)

3.4 Statistical Difference based Grouping Test (SD-G)

The Scott-Knott test [41] was proposed to rank and group means of a set of variables. However, using this test requires that input data fulfills the ANOVA assumptions, i.e., normality of distribution, homogeneity of variance, and independence of observations [22]. To relax this prerequisite, the Scott-Knott ESD V1 [46] was proposed, which applies log-transformation to correct the non-normal distribution of the input data. However, as reported in [22], the log-transformation may not make the data conform to the normal distribution in some cases and even might lead to a negative impact on the accuracy of the Scott-Knott test [41]. Given the negative impact of the log-transformation employed in the Scott-Knott ESD V1, the Scott-Knott ESD V2 [47] does not, anymore, apply any data pre-processing (e.g., using the log-transformation) to relax the normal distribution prerequisite. The non-parametric Scott-Knott ESD test [45] was; however, proposed to handle data with non-normal distributions. Considering that some of our data does not fulfill the ANOVA assumptions, we, thus, opted for the non-parametric Scott-Knott ESD test. However, we observed that in some cases, variables with negligible difference were grouped into different groups while variables with non-negligible difference were grouped into the same group. For example, Fig. 6 (a) presents a simple example of applying the non-parametric Scott-Knott ESD test, with the inputs shown in Table 7. We can see that variables V2 and V4 (also variables V6 and V8) are grouped into different

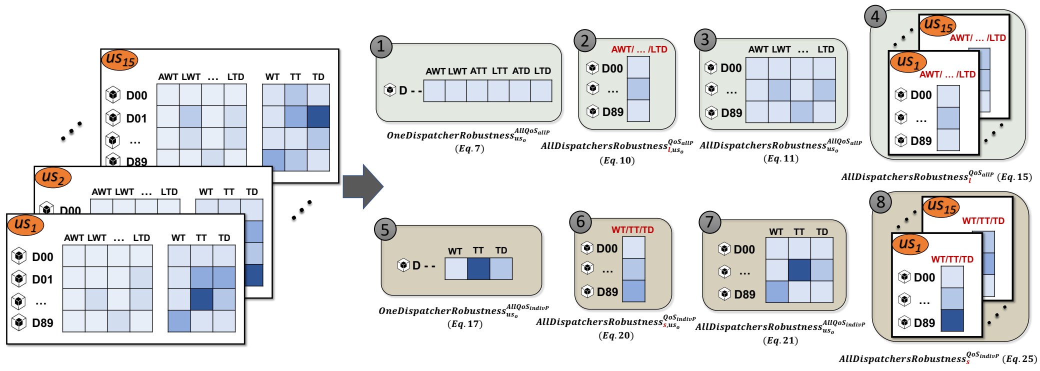


Fig. 5. Quantification of QoS_{allP} (①-④) and QoS_{indivP} (⑤-⑧) robustness. A grid represents the quantified robustness of a dispatcher under a specific uncertain situation in terms of a specific QoS_{allP} metric (Eq. 5) or QoS_{indivP} metric (Eq. 6). For QoS_{allP} robustness, ① shows the robustness of one dispatcher under a specific uncertain situation in terms of all the QoS_{allP} metrics; ② and ③ represents all dispatchers' robustness under a specific uncertain situation in terms of a specific QoS_{allP} metric, and all the QoS_{allP} metrics, respectively; ④ reflects all dispatchers' robustness in terms of a specific QoS_{allP} metric. Similarly, ⑤ - ⑧ show the corresponding QoS_{indivP} robustness.

groups, but the difference between them is negligible based on the Cliff's Delta effect size, i.e., $|d| = 0.111$ (negligible) for both variable pairs. The Cliff's Delta estimate [14] is calculated with:

$$d = \frac{\sum_{i=1}^{N_1}\sum_{j=1}^{N_2}\left([x_i > y_j] - [x_i < y_j]\right)}{N_1 \times N_2} \tag{2}$$

where $[\cdot]$ is the *Iverson bracket*, which is 1 when the statement $\cdot$ is true, and 0 otherwise. N_1 and N_2 are the observation sizes of the two variables being compared (e.g., the observation size of each variable is three in Table 7), respectively, and x_i and y_j are the ith and jth observations of the two variables, respectively. Correspondingly, the Cliff's Delta effect size can be obtained using the thresholds provided by Romano et al. [39], i.e., *negligible* ($|d| < 0.147$), *small* ($0.147 \leq |d| < 0.33$), *medium* ($0.33 \leq |d| < 0.474$), and *large* otherwise. Take the variable pair V2 and V4 for instance, the Cliff's Delta effect size is calculated as: $\left|\frac{(-1)+(-1)+(-1)+(-1)+1+1+1}{3\times 3}\right| = 0.111$ (negligible), the corresponding dominance matrix is shown in Table 8.

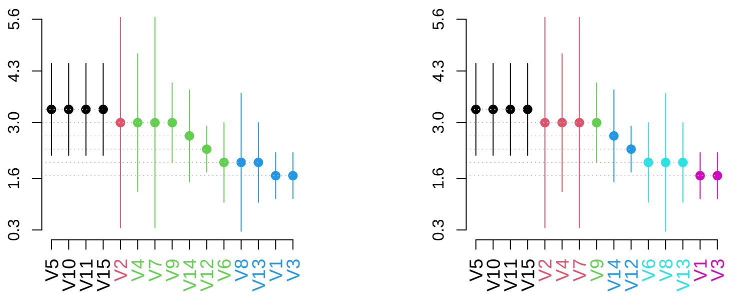
Moreover, V13 and V1 have non-negligible difference (i.e., $|d| = 0.222$ (small)), but were grouped into the same group. Similarly, V7 and V9 (i.e., $|d| = 0.222$ (small)), also V9 and V14 (i.e., $|d| = 0.222$ (small)), V12 and V6 (i.e., $|d| = 0.222$ (small)), with non-negligible differences, were grouped into the same group. More obviously, variable pairs V6 and V13, and V2 and V7, are the pairs that two variables have the same effect (i.e., distributions of the three observations are the same, $|d| = 0$), but were grouped into different groups. Thus, we further investigated the partition function of the non-parametric Scott-Knott ESD test, and found that the use of the Kruskal-Wallis chi-squared statistics to identify a partition that maximizes the median values between variables might put variables with negligible differences into different groups. In addition, they performed a follow-up comparison using the Cliff's Delta effect size, if the follow-up comparison of the variable pair has non-negligible difference then the group is split into two groups based on the identified partition, otherwise keep the one group. However, we found that, in their implementation, for the follow-up comparison, they only compare the first variable sample and the last variable sample rather than each variable pair. And if the difference between the first and last variables is negligible, the group is not split into two. Doing so might put variables into the same group (e.g., V13 and V1 in our example), thereby incorrectly indicating non-negligible differences.

Table 7. A simple set of running samples with 15 variables (treatments), each of which has three observations.

V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	V11	V12	V13	V14	V15
2	1	2	2	4	3	1	4	3	4	4	2	2	2	4
2	6	2	5	4	2	6	1	4	4	4	3	3	4	4
1	2	1	2	2	1	2	1	2	2	2	2	1	2	2

Table 8. Dominance matrix for the sample data of the variable pair V2 and V4.

	2	2	5
1	-1	-1	-1
2	0	0	-1
6	1	1	1



(a) Non-parametric Scott-Knott ESD test. (b) *SD-G* test.

Fig. 6. The partition of the variables in Table 7 with the non-parametric Scott-Knott ESD test and the *SD-G* test. The x-axis shows the sorted variables according to their observations. The same color of the adjacent variables indicates the same group. Each dot represents the average observations.

To this end, we developed our own grouping method (Algorithm 2) to fix the limitation in the non-parametric Scott-Knott ESD. Assuming a set of treatments $T = \{T_e | e = 1, 2, ..., nt\}$, where T_e is the eth treatment; nt is the number of treatments, same as the Scott-Knott ESD V2, we first sort the treatment means (Line 1). The sorted treatments are $ST = \{ST_f | f = 1, 2, ..., nt\}$, where ST_f is the fth treatment after sorting. Then, we use a sliding window to find the partition where all the treatment pairs that fall in the same group are statistically similar, i.e., the statistical difference of each pair in the group are negligible. Lines 2-9 define a function for the comparison of all the treatment pairs in the sliding window, we use the Cliff's Delta effect size [14], which does not require the ANOVA assumptions, to analyse the difference between two treatments. The comparison function returns TRUE when each treatment pair in all the treatments has negligible difference, otherwise it returns FALSE.

Lines 10-16 define a function that calls the comparison function for partitioning, which returns the number of treatments grouped by the sliding window each time. The sliding window with an initial size of 2 will be used to group the sorted treatments, if all the treatment pairs located in the sliding window have negligible difference, then the algorithm increases the sliding window size and performs the comparison again. If the non-negligible difference is found in the sliding window or the sliding window is bigger than the number of the treatments to be grouped, then the algorithm returns to the previous sliding window size. Lines 17-26 group all the treatments into several groups. The sliding window starts from the left side of the sorted treatments (i.e., $BeginIndex = 1$, nt is the total number of treatments) and performs the partitioning by calling the *Partition* function. Once all the treatments are grouped, we can get the ranked groups $G = \{(ST_f, r_f)|f = 1, 2, ..., nt\}$, where r_f is the group/rank number of ST_f (e.g., $r_1 = r_2 = 1$ means both ST_1 and ST_2 are grouped into the first group). For the running example (with the inputs in Table 7), results of the *SD-G* test are presented in Fig. 6 (b), from where we can see that the grouping problems in the non-parametric Scott-Knott ESD test (Fig. 6 (a)) are resolved with our *SD-G* test.

3.5 Parameter Settings and Experiment Execution (Step 3 in Fig. 2)

We experimented with the 90 dispatchers. To ensure a fair comparison, we used the same settings and the same traffic profiles for each dispatcher. The building configuration is the same as we used to generate the uncertain traffic profiles (see Section 3.1.4). We ran all the generated uncertain traffic profiles (see Section 3.1.5) corresponding to the 15 uncertain situations. After all the traffic profiles were executed with all the dispatchers, we performed the robustness analysis based on the QoS_{allP} and QoS_{indivP} generated by each dispatcher. To answer the RQs, we analyzed all the QoS_{allP} and QoS_{indivP} produced by each dispatcher, and conducted the *SD-G* test (see Section 3.4) from various aspects (see Section 3.3), the analysis results will be presented in Section 4.

4 EXPERIMENT RESULTS AND ANALYSIS

4.1 Results of RQ1

4.1.1 RQ1.1: Comparing all the 90 dispatchers in terms of QoS_{allP} robustness. We conducted the *SD-G* test with Eq. 8 to rank all the 90 dispatchers in terms of each QoS_{allP} robustness for *LunchPeak* and *UpPeak*. The results are provided in Fig. 14 in Appendix B for reference. In general, we observe that, for *LunchPeak*, the 89 generated versions exhibit worse or similar robustness than the original dispatcher in terms of each QoS_{allP} metric. For *UpPeak*, all the generated dispatchers performed less robust than or similar to the original dispatcher in terms of *AWT*, *ATT*, *ATD* and *LTD*, while some generated dispatchers performed more robust than the original one in terms of *LWT* and *LTT*. One possible explanation could be that some changes to the original dispatcher may favor the longest waiting and transit times among all the passengers, which, however, may also degrade its robustness in terms of other QoS_{allP} metrics. For example, for *UpPeak*, D58 performed quite robust in terms of *LWT*, while performed the least robust in terms of *AWT*.

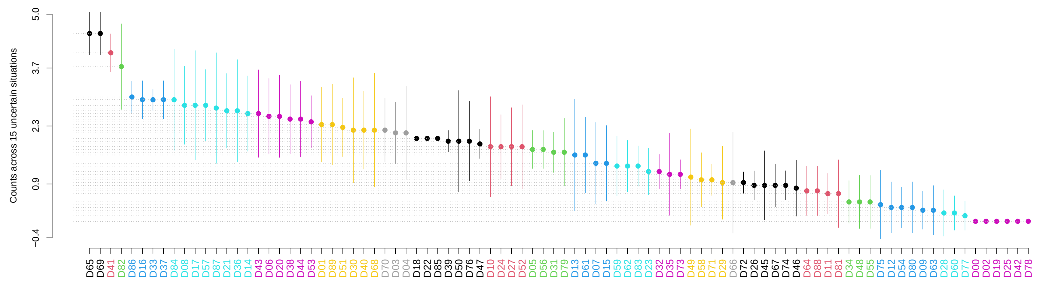
To compare the robustness of all the dispatchers in terms of all the QoS_{allP} metrics, we first quantified the robustness of each dispatcher in terms of all the QoS_{allP} metrics under each uncertain situation with Eq. 7. Results are summarized in the left two subplots in Fig. 15 in Appendix B. We observe that most of the generated dispatchers are less robust than the original one when dealing with passengers' uncertainties. In addition, some generated dispatchers exhibit various extents of

Algorithm 2: Statistical Difference based Grouping Algorithm (SD-G)

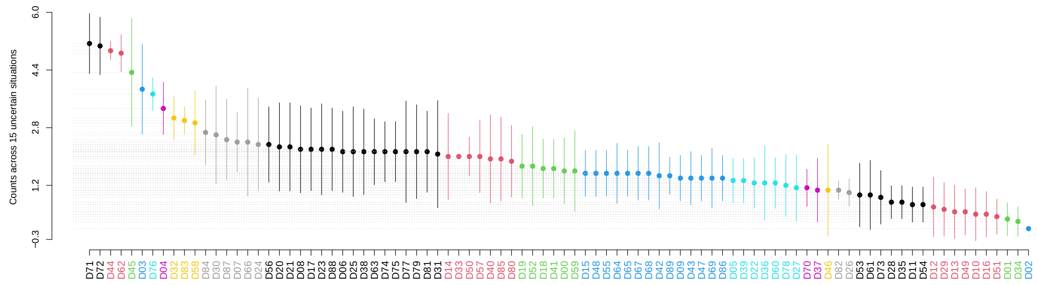
```
Input: Treatments $T$
Output: Groups $G$
1  /* Sorting the treatment means */
2  $ST \leftarrow Sort(T)$
3  /* Partitioning with sliding window */
4  Function CompareAll($ST$, $BeginIndex$, $WinSize$):
5      $negl \leftarrow$ TRUE
6      for $y \leftarrow BeginIndex$ to $(BeginIndex + WinSize - 2)$ do // Compare all pairs in the sliding window
7          for $z \leftarrow BeginIndex + 1$ to $(BeginIndex + WinSize - 1)$ do
8              $ef \leftarrow CliffsDelta(ST_y, ST_z)$
9              if $ef \neq$ "$negligible$" then
10                 $negl \leftarrow$ FALSE
11     return $negl$
12 Function Partition($ST$, $BeginIndex$, $nt$):
13     $WinSize \leftarrow 2$
14     while CompareAll($ST, BeginIndex, WinSize$) do // Increase window size when the difference of each pair is negligible
15         $WinSize \leftarrow WinSize + 1$
16         if $WinSize > (nt - BeginIndex + 1)$ then
17             break
18     return $WinSize - 1$
19 $rank \leftarrow 0$
20 while $BeginIndex < nt$ do
21     $num \leftarrow$ Partition($ST, BeginIndex, nt$)
22     $rank \leftarrow rank + 1$
23     for $w \leftarrow BeginIndex$ to $(BeginIndex + num - 1)$ do
24         $r_w \leftarrow rank$
25     $BeginIndex \leftarrow BeginIndex + num$ // Update the position of the sliding window
26 if $BeginIndex = nt$ then // If there is only one treatment left, then itself forms a group
27     $rank \leftarrow rank + 1$
28     $r_{nt} \leftarrow rank$
```

robustness across the two templates. For instance, D25 performed very robust in *LunchPeak*, while showed poor robustness in *UpPeak*. It tells that the QoS_{allP} robustness of a dispatcher might be specific to templates, i.e., related to passenger traffics.

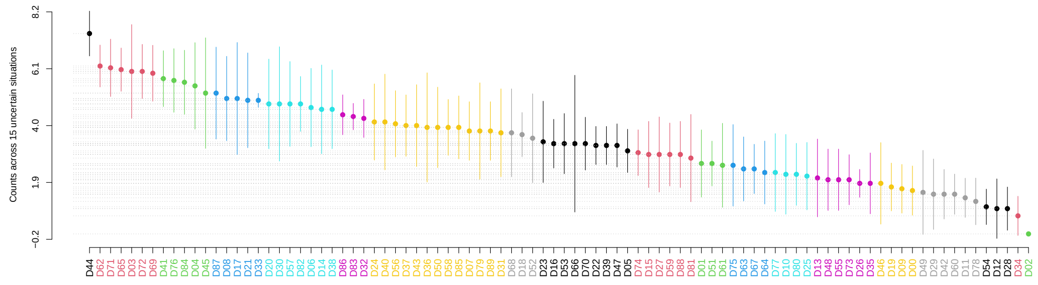
To further compare the QoS_{allP} robustness of the 90 dispatchers across the 15 uncertain situations, we then conducted a ranking with Eq. 9 based on the counts visualized in the left two subplots in Fig. 15. The template specific ranking of all the 90 dispatchers are shown in Fig. 7 (a) and (b). **For *LunchPeak*, 84 out of 89 (94%) generated versions are less robust than the original dispatcher in terms of QoS_{allP} metrics**; 5 out of 89 (6%) generated versions are statistically similar to the original dispatcher in terms of QoS_{allP} metrics; and D65 and D69 are the least robust against passengers' uncertainties. **For *UpPeak*, 41 out of 89 (46%) generated versions performed worse than the original dispatcher;** 43 out of 89 (48%) generated versions performed better than the original dispatcher; 5 out of 89 (6%) generated versions have statistically similar robustness as the original dispatcher; and D71 and D72 are the least robust ones. For both templates, D02 shows relatively better robustness than the other generated versions. The reason why some generated versions performed better than the original dispatcher could be that the introduced change makes the dispatcher improve specific situations. For instance, one change is about favoring the parking of the elevators in the base floor, which has a positive impact on the *UpPeak* profile, but not for other situations.



(a) Template specific ranking - *LunchPeak*.



(b) Template specific ranking - *UpPeak*.



(c) Overall ranking on both templates.

Fig. 7. The *SD-G* ranking of the 90 dispatchers across the 15 uncertain situations in terms of all the QoS_{allP} metrics - RQ1.1. The x-axis shows the generated (D01-D89) and the original (D00) dispatchers sorted according to their counts under all the uncertain situations in terms of all the QoS_{allP} metrics, whereas the same color of the adjacent dispatchers indicates the same group, implying that the difference of the robustness of these dispatchers across the 15 uncertain situations in terms of all the QoS_{allP} metrics are negligible. Each dot represents the average count across the 15 uncertain situations on each sample, and the length of each line indicates twice of the standard deviation of each sample.

To compare the overall robustness of the 90 dispatchers on both templates, we conducted ranking again with Eq. 9, based on the quantified robustness of each generated version and the original dispatcher on both templates (i.e., summing the counts visualized in the left two subplots in Fig. 15), as shown in Fig. 7 (c). **Overall, 75 out of 89 (84%) generated versions are worst than the original dispatcher in terms of QoS_{allP} robustness**; 11 out of 89 (12%) dispatcher versions are more robust than the original dispatcher; and 3 out of 89 (3%) generated versions have statistically similar robustness as the original dispatcher. We carefully studied the 11 dispatcher versions that showed better robustness than the original dispatcher and confirmed with the domain experts that there is no false positive in the observation. The reason for the 11 generated versions performing better than the original one could be two-fold. First, the generated versions affect certain parameters that better fit in the selected building. Second, some changes favour *UpPeak* and *LunchPeak* traffic templates, but could negatively impact other types of traffic templates (e.g., *DownPeak*) as well as the dispatcher in a non-simulated environment, which we did not test in this empirical study. We have reported these 11 versions to Orona for them to investigate which change made the dispatcher more robust during *LunchPeak* and *UpPeak*. The investigation results tell us that most of these changes are related to threshold values that activate certain functionalities more frequently, helping the dispatcher behave better under certain circumstances.

Conclusion for RQ1.1: 1) The QoS_{allP} robustness of the dispatchers varies from template to template; **2)** Overall, most of the 89 generated versions (84%) are less robust than the original dispatcher in terms of QoS_{allP} metrics, implying that **a majority of changes reduce the original dispatcher's QoS_{allP} robustness against passengers' uncertainties**.

4.1.2 *RQ1.2: Comparing Uncertain Situations in terms of QoS_{allP} Robustness.*

To investigate the impact of the different uncertain situations on the QoS_{allP} robustness of the 90 dispatchers, we quantified the QoS_{allP} robustness of all the 90 dispatchers under each uncertain situation (see Eq. 10 and Eq. 11) on each template. Results are summarized in Table 12 in Appendix B.

Overall, we observed that the difference in the impact of the uncertain situations on the QoS_{allP} robustness of the 90 dispatchers, in most cases, is relatively small. For instance, the difference in the QoS_{allP} robustness in terms of *AWT*, under the 15 uncertain situations, during *LunchPeak*, is small as indicated by the counts ranging from 12 to 20, out of 90, which is less than (20-12)/90 = 9% of difference across the uncertain situations. However, there are exceptions. For instance, for *LTT* under *UpPeak*, the counts range from 9 to 77, which is 76% of difference across all the 15 uncertain situations. In addition, the maximum count, i.e., 77 for *usC-L* in *UpPeak*, reaches the 86% of the total. When comparing *LunchPeak* and *UpPeak*, we can observe that **the QoS_{allP} robustness of the 90 dispatchers under the 15 uncertain situations during *UpPeak* fluctuates more than during *LunchPeak***, implying that the impact of the uncertain situations on the QoS_{allP} robustness of the 90 dispatchers varies for different templates.

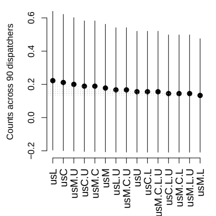
To further compare the impact of the uncertain situations on the QoS_{allP} robustness of the 90 dispatchers, we ranked all the uncertain situations with Eq. 12 for *LunchPeak* and *UpPeak*. Results are shown in Fig. 8 (a)-(l). We can observe, from Fig. 8 (a)-(f), that, for *LunchPeak*, the 15 uncertain situations were ranked into either one or two groups (except for *LTT* having *usL-U* classified as the third group). This indicates that most uncertain situations do not have many differences in terms of influencing the QoS_{allP} robustness across the 90 dispatchers. However, for *UpPeak*, as shown in Fig. 8 (j), in terms of *LTT*, the uncertain situations were categorized into four groups. A plausible explanation might be that the traffic flow during *UpPeak* is mostly incoming. As we can see from Fig. 4 that the majority of the passenger activities from 08:30 to 09:30 are incoming (the green part), and the incoming activities peaked around 09:00. It means that most people come to work around 9:00 am and take elevators from the entrance floor to different floors of the office building. This pattern might lead to a higher transit time for those who work on top floors, as the elevator needs to frequently unload passengers at various floors. This is prominent for transit time related QoS_{allP} metrics, especially LTT. This may happen due to two main reasons. Firstly, the dispatchers we investigated were configured to optimize passengers' waiting times. However, LTT does not consider the waiting time of passengers. Secondly, the type of traffic pattern may lead to not favoring some QoS_{allP} metrics. In this case, during up peak, elevators tend to travel upwards with cabins full of people. Conversely, unlike during lunch peak patterns, when traveling downwards during an up peak, the cabins travel empty. When arriving on the base floor, people traveling upward enter to different elevators. However, in a conventional elevator [9], which is the type of elevators we use, passengers heading to the same floors are not grouped. Thus, many elevators travel to the same floor, resulting in long transit times for passengers traveling to the top floors.

To compare the impact of the 15 uncertain situations on the robustness of the dispatchers in terms of all the QoS_{allP} metrics, we performed ranking with Eq. 13 on each template. Results of the ranking for *LunchPeak*, *UpPeak* and *Overall* are shown in Fig. 8 (m)-(o), respectively. From the overall ranking, we can observe that, in general, *usC*, *usM-C* and *usM* have relatively large impact on the QoS_{allP} robustness of the 90 dispatchers, while *usC-L-U*, *usU*, *usL-U* and *usM-L-U* have relatively small impact on the QoS_{allP} robustness. It indicates that uncertain factors *Capacity Factor*, *Mass*, and their interactions have more impact on the QoS_{allP} robustness of the 90 dispatchers, whereas *Unloading Time* and its interactions with some other uncertain factors (e.g., *Loading Time*) have relatively less impact on the QoS_{allP} robustness of the 90 dispatchers. That's because a lower *Capacity Factor* may lead a passenger not to enter in an elevator and therefore need to wait to another one. In the case of the *Mass*, the elevator may not be prepared to lift an additional passenger that was in the dispatching plan of the algorithm if the *Mass* of the passengers that are inside the elevator is higher than expected. In both of these cases, the passenger would need to wait for another elevator, which takes much longer time than typical *Loading* and *Unloading* times (in a few seconds).

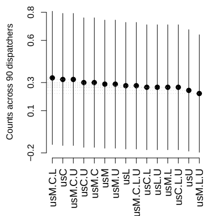
Based on the QoS_{allP} robustness of the 90 dispatchers under each uncertain situation, we further investigated the correlation between the number of uncertain factors and the QoS_{allP} robustness of the 90 dispatchers, e.g., 2-way vs. 3-way. We noticed that interactions of a higher number of uncertain factors do not necessarily lead to poorer QoS_{allP} robustness. For example, as we can see from Fig. 8 (a) and (b), interactions between more uncertain factors have similar impacts as interactions between fewer uncertain factors on the QoS_{allP} robustness across the 90 dispatchers. We can also observe, from Fig. 8, that some uncertain situations caused by single uncertain factor are ranked into the first group (e.g., *usC* and *usM* in Fig. 8 (m)), while some are ranked into the last group (e.g., *usU* in Fig. 8 (n) and (o)), which is the same for other cases such as interactions between two or three uncertain factors. We therefore recommend elevator engineers to systemically test dispatchers' robustness against passengers' uncertainties under each uncertain situation, which can provides engineers insights to design uncertain situation-oriented robustness optimization strategy.

Conclusion for RQ1.2: 1) The impact of the uncertain situations on dispatchers' QoS_{allP} robustness varies from template to template. For instance, the 15 uncertain situations exhibit 15% of the difference in the impact on the QoS_{allP} robustness for *LunchPeak* and 25% for *UpPeak*; **2)** Overall, uncertain factors *Capacity Factor*, *Mass*, and their interactions have relatively

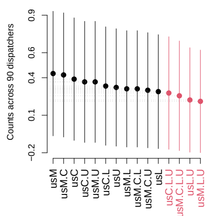
[9]For a conventional elevator, its dispatcher is not aware of the final destinations of its passengers.



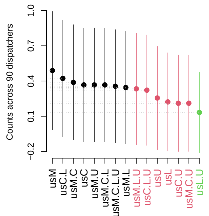
(a) *AWT - LunchPeak.*



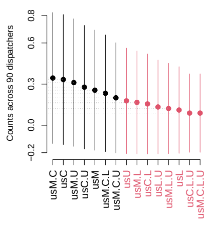
(b) *LWT - LunchPeak.*



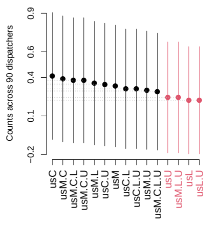
(c) *ATT - LunchPeak.*



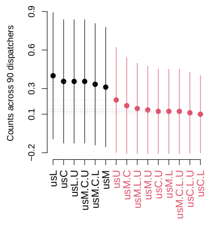
(d) *LTT - LunchPeak.*



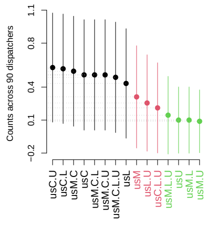
(e) *ATD - LunchPeak.*



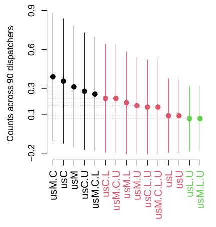
(f) *LTD - LunchPeak.*



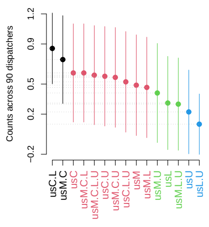
(g) *AWT - UpPeak.*



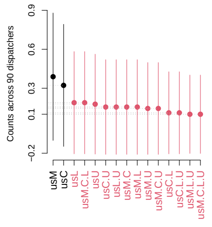
(h) *LWT - UpPeak.*



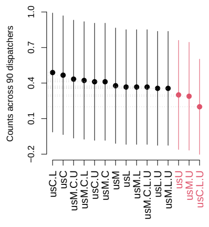
(i) *ATT - UpPeak.*



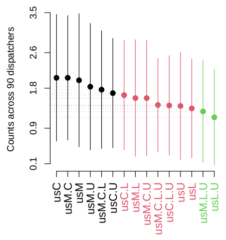
(j) *LTT - UpPeak.*



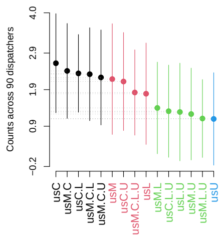
(k) *ATD - UpPeak.*



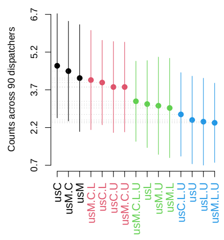
(l) *LTD - UpPeak.*



(m) Ranking on *LunchPeak.*



(n) Ranking on *UpPeak.*



(o) Ranking on both templates.

Fig. 8. The *SD-G* ranking of the 15 uncertain situations across the 90 dispatchers in terms of each QoS_{allP} (Fig. 8 (a)-(l)) and all the QoS_{allP} (Fig. 8 (m)-(o)) metrics - RQ1.2. An x-axis shows the uncertain situations sorted according to their counts in terms of a specific QoS_{allP} (see *Count* columns of the six QoS_{allP} in Table 12) or all the QoS_{allP} (see column *Overall* in Table 12), whereas the same color of the adjacent uncertain situations indicates the same group, implying that the robustness difference across the 90 dispatchers under these uncertain situations are negligible. A lower count (y-axis) indicates a higher robustness under a specific uncertain situation. Each dot represents the average count across the 90 dispatchers on each sample, and the length of each line indicates twice of the standard deviation of each sample.

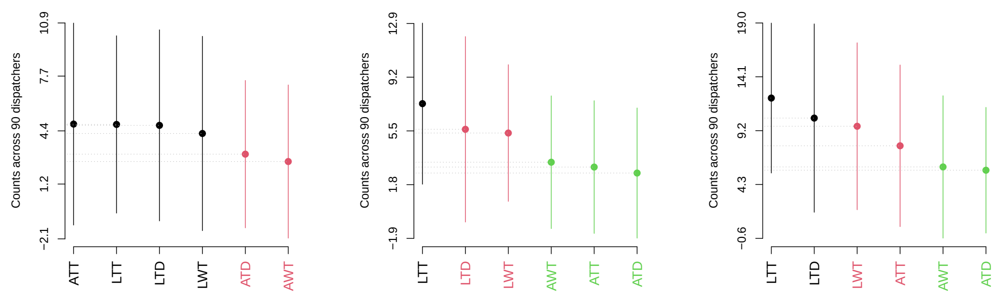
large impact on the QoS_{allP} robustness of the 90 dispatchers, whereas *Unloading Time* and its interactions with some other uncertain factors (e.g., *Loading Time*) have relatively less impact on the QoS_{allP} robustness across the 90 dispatchers; **3)** Uncertain situations with more uncertain factors' interactions do not necessarily have more negative impact on the QoS_{allP} robustness.

4.1.3 RQ1.3: Rank QoS_{allP} metrics. To investigate to which degree each QoS_{allP} metric is impacted by passengers' uncertainties, we quantified the robustness of all the 90 dispatchers against passengers' uncertainties in terms of each QoS_{allP} metric with Eq.15 on *LunchPeak* and *UpPeak*. The sorted QoS_{allP} metrics on the two templates for all the 90 dispatchers are shown in Table 9. We can see that for *LunchPeak*, the 90 dispatchers have the highest count in terms of *ATT*, whereas they have the lowest count in terms of *AWT*. For *UpPeak*, the 90 dispatchers are least robust in terms of *LTT*, whereas they are relatively more robust in terms of *ATD*.

Table 9. QoS_{allP} robustness of the dispatchers - RQ1.3. For each QoS_{allP} metric, there are, in total, 90 (dispatchers) × 15 (uncertain situations) = 1350 p-values. Each *Sorted QoS* cell tells the count calculated with Eq.15. A lower count indicates a higher robustness.

Template	**Sorted QoS_{allP} Metrics**						**Total Count**
LunchPeak	ATT (433)	LTT (431)	LTD (426)	LWT (382)	ATD (270)	AWT(230)	2172
UpPeak	LTT (664)	LTD (505)	LWT (482)	AWT (301)	ATT (271)	ATD (234)	2457

We further investigated the difference in the QoS_{allP} robustness across all the 90 dispatchers on each template by conducting a ranking with Eq. 16. Results are shown in Fig. 9 (a) and (b). We also performed the overall ranking by combining the two templates. Results are shown in Fig. 9 (c). Fig.



(a) Template specific ranking - *LunchPeak.*

(b) Template specific ranking - *UpPeak.*

(c) Overall ranking.

Fig. 9. The *SD-G* ranking of the six QoS_{allP} metrics across the 90 dispatchers - RQ1.3. The x-axis shows the QoS_{allP} metrics sorted according to their counts, whereas the same color of the adjacent QoS_{allP} metrics indicates the same group (e.g., *ATD* and *AWT* in Fig. 9 (a) are in the same group), implying that the difference of the robustness across the 90 dispatchers in terms of these QoS_{allP} metrics are negligible. A lower count (y-axis) indicates a higher robustness in terms of a specific QoS_{allP} metric. Each dot represents the average count across the 90 dispatchers on each sample, and the length of each line indicates the twice of the standard deviation of each sample.

9 (a) and (b) show that for *LunchPeak*, the difference of the robustness across the 90 dispatchers

in terms of *ATT*, *LTT*, *LTD* and *LWT* are negligible, and the 90 dispatchers have relatively better robustness in terms of *ATD* and *AWT* than *ATT*, *LTT*, *LTD* and *LWT*. For *UpPeak*, *LTT* is the only QoS_{allP} included in the group at the first place, indicating that the 90 dispatchers are the least robust in terms of *LTT*. A possible reason is that *LTT*, being the longest transit time, is relatively more sensitive to passengers' uncertainties during *UpPeak*. Since during *UpPeak*, passengers with diverse destinations often take an elevator at the entrance floor to their respective working floors. As a result, the elevator needs to stop frequently to unload passengers on different floors while going upwards, which consequently increases the transit time of passengers with the top floor as their destination, making the longest transit time relatively more susceptible to passengers' uncertainties. In addition, we can find that *AWT*, *ATT* and *ATD* are ranked into the last group, which means that the 90 dispatchers are statistically similar and relatively more robust in terms of the average time-related QoS_{allP} metrics (i.e., *AWT*, *ATT* and *ATD*) during *UpPeak*. From the overall ranking (Fig. 9 (c)), we can first observe that the average counts across all the dispatchers in terms of the longest time-related QoS_{allP} metrics are higher than that of the average time-related QoS_{allP} metrics, implying that the average time-related QoS_{allP} metrics are relatively impacted less than the longest time-related QoS_{allP} metrics. Second, we can observe that *LTT* and *LTD* are ranked into the first group, *LWT* and *ATT* are ranked into the second group, and *AWT* and *ATD* are also ranked into the last group. A possible explanation is that the dispatchers we used is intended to optimize average waiting time of passengers, which is also considered by *ATD*. Therefore, their robustness is less affected by passengers' uncertainties.

Conclusion for RQ1.3: 1) *AWT* and *ATD* are ranked into the last group in all the three rankings, implying that the 90 dispatchers have statistically similar and relatively better robustness against passengers' uncertainties in terms of *AWT* and *ATD* than the other QoS_{allP} metrics. This might be because the dispatchers we used mainly focus on optimizing average waiting time; **2)** *LTT* is ranked into the first group in all the three rankings, which tells that the 90 dispatchers are relatively less robust in terms of *LTT*; **3) Overall, the average time-related QoS_{allP} metrics (i.e., *AWT*, *ATT* and *ATD*) are relatively less impacted, by passengers' uncertainties, than the longest time-related QoS_{allP} metrics (i.e., *LWT*, *LTT* and *LTD*).**

4.2 Results of RQ2

4.2.1 RQ2.1: Comparing all the 90 dispatchers in terms of QoS_{indivP} robustness. We conducted the *SD-G* test to rank all the 90 dispatchers in terms of each QoS_{indivP} robustness (Eq. 18). Results are provided in Fig. 16 in Appendix B for reference. In summary, we observed that most of the generated versions performed less robust than the original dispatcher in terms of *WT* and *TD*, while more generated versions exhibit better robustness than the original dispatcher in terms of *TT*, during *LunchPeak*. Conversely, for *UpPeak*, a majority of the generated versions performed less robust than the original dispatcher in terms of *TT*, while there are relatively more generated versions that performed more robust than the original dispatcher in terms of *WT* and *TD*. These observations tell that **the robustness of the dispatchers in terms of a specific QoS_{indivP} metric is also related to the traffic template**. For example, D65, in terms of *WT*, is the most robust during *UpPeak*, while the least robust during *LunchPeak*. Similarly D77 shows the best robustness in terms of *TT* on *LunchPeak*, which is opposite on *UpPeak*.

We compare the 90 dispatchers in terms of all the QoS_{indivP} metrics by first quantifying the robustness of each dispatcher in terms of all the QoS_{indivP} metrics under each uncertain situation (Eq. 17) on each template. Results are summarized in the right two subplots in Fig. 15 in Appendix B.

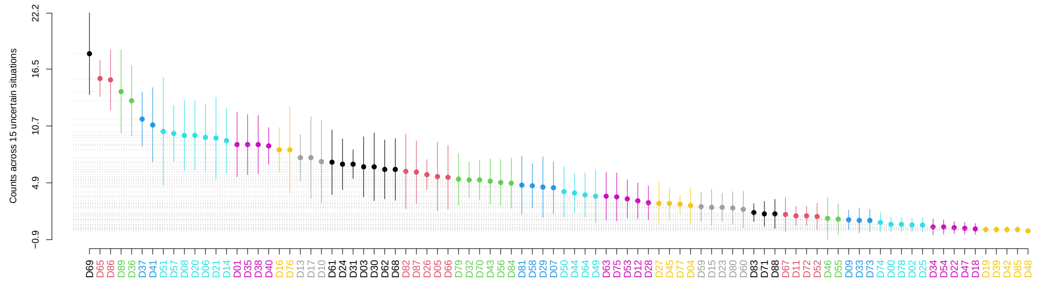
In summary, we observed that some dispatchers were less robust under most uncertain situations, whereas some dispatchers were very robust under almost all the uncertain situations. In addition, we also observed that some dispatchers show completely different robustness on the two templates. For example, D65 was less robust on *LunchPeak*, while performing very robust on *UpPeak*, implying that the QoS_{indivP} robustness of the dispatchers is also related to the templates. A potential explanation for this might be that some specific dispatcher versions provoke an improvement in specific traffic templates. For instance, one generated version might constantly send an elevator to the base floor benefiting the *UpPeak* traffic template. Another explanation could be that a dispatcher relies on different parameters. For instance, a parameter might be related to the number of parked elevators to be sent to the base floor. A change in the code may influence one parameter, which favors a particular dispatcher under certain circumstances.

We further ranked the QoS_{indivP} robustness of all the 90 dispatchers across the 15 uncertain situations for each template with Eq. 19. Results are shown in Fig. 10 (a) and (b), respectively. **For *LunchPeak*, 75 out of 89 (84%) generated versions performed less robust than the original dispatcher in terms of QoS_{indivP} robustness**; 10 out of 89 (11%) generated versions (e.g., D34 and D19) performed better than the original dispatcher; 4 out of 89 (4%) generated versions (i.e., D74, D78, D02 and D25) have statistically similar QoS_{indivP} robustness as the original dispatcher; and D69 performed the least robust. **For *UpPeak*, 55 out of 89 (62%) generated versions performed less robust than the original dispatcher**; 24 out of 89 (27%) generated versions performed better than the original dispatcher; 10 out of 89 (11%) generated versions have statistically similar QoS_{indivP} robustness as the original dispatcher; whereas D44, D71 and D72 performed the least robust against passengers' uncertainties.

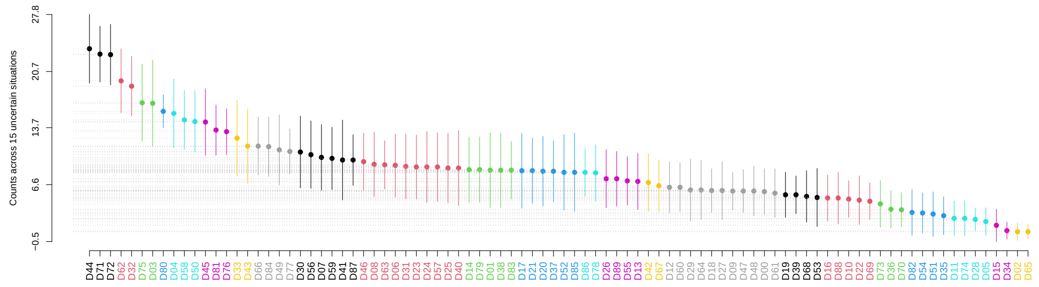
To further compare the overall robustness of the 90 dispatchers on both templates, we conducted ranking (Eq. 19) based on the quantified robustness of each dispatcher under each uncertain situation in terms of all the QoS_{indivP} metrics on both templates (i.e., combining the QoS_{indivP} robustness shown in the right two subplots in Fig. 15 together), as shown in Fig. 10 (c). **Overall, 74 out of 89 (83%) generated versions performed worse than the original dispatcher in terms of QoS_{indivP} robustness**; 11 out of 89 (12%) generated versions performed better than the original dispatcher; and 4 out of 89 (4%) generated versions have statistically similar QoS_{indivP} robustness as the original dispatcher.

Conclusions for RQ2.1: **1)** The QoS_{indivP} robustness of the dispatchers is also related to the templates, thus we recommend elevator engineers to assess a new version with as diverse templates as possible to gain more comprehensive insights of its quality before deploying; **2) Overall, most of the generated versions (83%) performed less robust than the original dispatcher in terms of QoS_{indivP} metrics**, implying that a majority of changes in code downgrade the original dispatcher's QoS_{indivP} robustness against passengers' uncertainties.

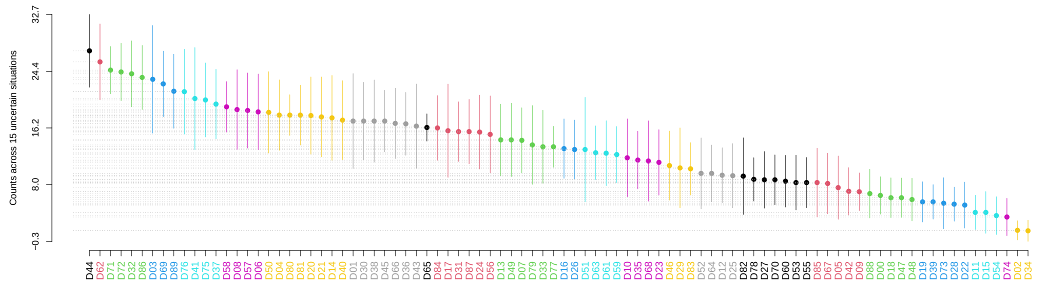
4.2.2 RQ2.2: Comparing Uncertain Situations in terms of QoS_{indivP} Robustness. We assess the impact of each uncertain situation on dispatchers' QoS_{indivP} robustness by quantifying the QoS_{indivP} robustness of all the dispatchers under each uncertain situation (see Eq. 20 and Eq. 21) for *LunchPeak* and *UpPeak*. Results are summarized in Table 13 in Appendix B. In summary, we observed that the 15 uncertain situations have different extents of impact on the QoS_{indivP} robustness of the 90 dispatchers. For instance, the difference in the impact of the uncertain situations on the QoS_{indivP} robustness in terms of *WT* during *LunchPeak* is small, as reflected by the counts ranging from 102 to 157, out of 900 ($(157 - 102)/900 = 6\%$). In contrast, the counts in terms of *TT* during *UpPeak* ranging from 7 to 492, which is more than 53% of difference across all the uncertain situations, implying that the uncertain situations have various impact on the QoS_{indivP} robustness of the 90 dispatchers



(a) Template specific ranking - *LunchPeak*.



(b) Template specific ranking - *UpPeak*.



(c) Overall ranking on both templates.

Fig. 10. The *SD-G* ranking of the 90 dispatchers across the 15 uncertain situations in terms of all the QoS_{indivP} metrics - RQ2.1. The x-axis shows the 89 generated versions (D01-D89) and the original dispatcher (D00) sorted according to their counts under all the uncertain situations in terms of all the QoS_{indivP} metrics, whereas the same color of the adjacent dispatchers indicates the same group, implying that the difference of the QoS_{indivP} robustness of these dispatchers across the 15 uncertain situations are negligible. Each dot represents the average count across the 15 uncertain situations on each sample, and the length of each line indicates twice of the standard deviation of each sample.

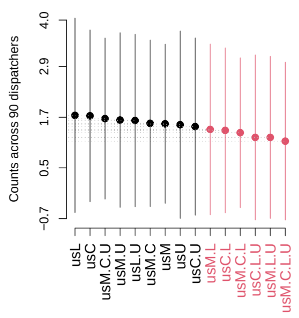
in terms of *TT* during *UpPeak*. When we compare *LunchPeak* and *UpPeak* (see Table 13 for detailed results), we observed that the counts for *UpPeak* are higher than for *LunchPeak*, and the counts fluctuate more during *UpPeak*. This is clearly reflected by column *Overall*, where the minimum and maximum counts are 284 and 615 (out of 2700) for *LunchPeak*, and 524 and 1137 for *UpPeak*, and $(615 - 284)/2700 = 12\%$ of differences across all the uncertain situations for *LunchPeak* and 23% for *UpPeak*. These observations imply that the uncertain situations have different impacts on the

QoS_{indivP} robustness of the 90 dispatchers for different templates, especially having a relatively large impact on the QoS_{indivP} robustness for *UpPeak*. In addition, we can observe that for both *LunchPeak* and *UpPeak*, *usL* has the highest count in terms of *WT*, and *usM-C* have the highest counts in terms of *TT*.

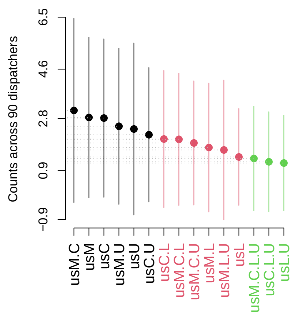
We further compare the impact of the uncertain situations on the QoS_{indivP} robustness by ranking them with Eq. 22 for each template. Results are summarized in Fig. 11 (a)-(f). The figure shows that the uncertain situations were ranked into either two or three groups for *LunchPeak*, and more than three groups for *UpPeak*. Especially, as shown in Fig. 11 (e), the 15 uncertain situations were categorized into nine groups in terms of *TT* for *UpPeak*. This indicates that the uncertain situations have more diverse impact on the QoS_{indivP} robustness for *UpPeak* than for *LunchPeak*, especially for *TT* during *UpPeak*, as shown in Table 13 ((492-7)/900=54% difference). This might be caused by the special traffic pattern during *UpPeak* in the office building, which has the following characteristics: the passenger activities are mostly incoming in the morning to the offices (as shown in Fig. 4). Therefore, many passengers register calls and wait for the elevators on the entrance floor. On the other hand, some do not even need to wait since they often hurry to the entrance floor, and luckily an elevator door is already open. For these passengers, their waiting time (*WT*) is 0. Moreover, as we observed, in most cases, each elevator loads passengers until there is no spare space. Then it closes the door and transits from the entrance floor to the different upper floors according to the passengers' requests. Correspondingly, the elevator needs to stop frequently to unload passengers, which makes *TT* relatively more sensitive to passengers' uncertainties than *WT* and *TD* during *UpPeak*. This is because the transit time of the passengers whose destinations are on higher floors is more likely affected by passengers' uncertainties.

To study the 15 uncertain situations in terms of all the QoS_{indivP} metrics, we performed ranking with Eq. 23. Results for *LunchPeak*, *UpPeak* and *Overall* (combing the two templates together) are shown in Fig. 11 (g), (h) and (i), respectively. Fig. 11 (g)-(i) show that for the two template-specific rankings (Fig. 11 (g) and (h)) and the overall ranking (Fig. 11 (i)), both *usM-C-L-U* and *usC-L-U* were categorized into the last group, whereas *usM* and *usC* were categorized into the first group. It indicates that uncertain factors *Mass* and *Capacity Factor* have more impact on the QoS_{indivP} robustness of the 90 dispatchers, while interactions between uncertain factors *Capacity Factor*, *Loading Time* and *Unloading Time*, and interactions between all the four uncertain factors have relatively small impact on the QoS_{indivP} robustness of the 90 dispatchers.

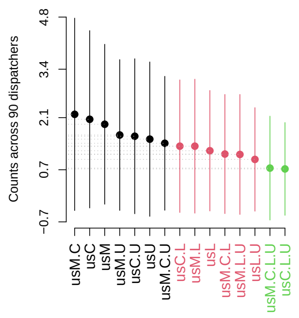
From the QoS_{indivP} robustness in Table 13 and Fig. 11, we also noticed that the QoS_{indivP} robustness of the 90 dispatchers is not correlated to the number of interacting uncertain factors, i.e., interactions of a higher number of uncertain factors do not necessarily worsen the QoS_{indivP} robustness. For instance, as we can observe from Table 13 and Fig. 11, some uncertain situations caused by a single uncertain factor (e.g., *usM* and *usC*) have relatively higher impact than the uncertain situations with more interacting uncertain factors (e.g., *usM-C-L-U*) on the QoS_{indivP} robustness. Recall from Section 3.1 that to generate an uncertain traffic profile for, e.g., *usM-C-L-U*, values of the four uncertain factors *Mass*, *Capacity Factor*, *Loading Time* and *Unloading Time* are varied around the recommended values. In other words, each of them might take a value that is higher or lower than their respective recommended values for each passenger, but the average values of the uncertain factors for all the passengers across each generated uncertain traffic profile are maintained as the recommended values, to be realistic. Therefore, it is worth mentioning that an uncertain situation such as *usM-C-L-U* is not a situation that has extreme values on *Mass*, *Capability Factor*, *Loading* and *Unloading Times*. Consequently, interactions of a higher number of uncertain factors does not necessarily worsen the QoS_{indivP} robustness as we observed.



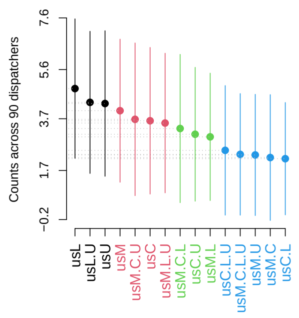
(a) *WT - LunchPeak.*



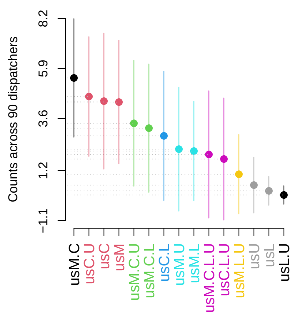
(b) *TT - LunchPeak.*



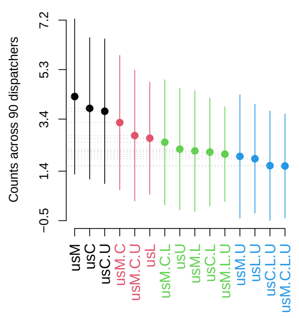
(c) *TD - LunchPeak.*



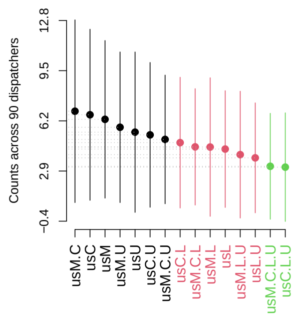
(d) *WT - UpPeak.*



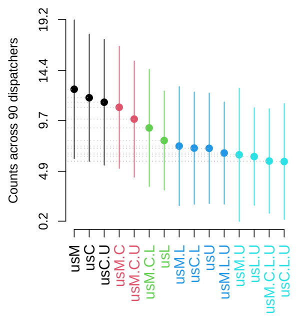
(e) *TT - UpPeak.*



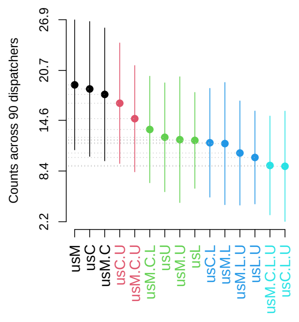
(f) *TD - UpPeak.*



(g) Template specific ranking - *LunchPeak.*



(h) Template specific ranking - *UpPeak.*



(i) Overall ranking on both templates.

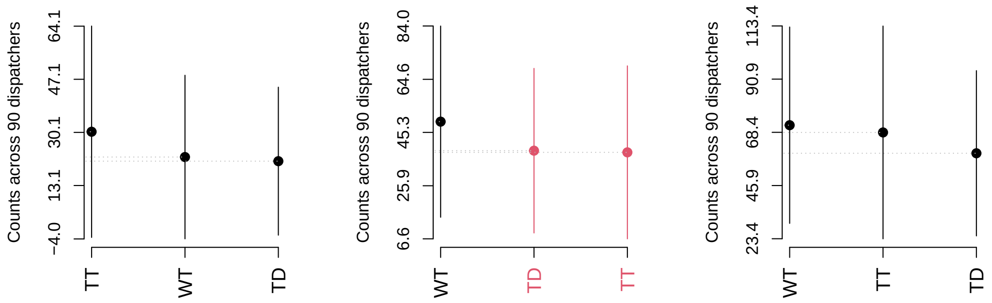
Fig. 11. The *SD-G* ranking of the 15 uncertain situations across the 90 dispatchers in terms of each QoS_{indivP} metric (Fig. 11 (a)-(f)) and all the QoS_{indivP} metrics (Fig. 11 (g)-(i)) - RQ2.2. An x-axis shows the uncertain situations sorted according to their counts in terms of a specific QoS_{indivP} (see *Count* columns of the three QoS_{indivP} metrics in Table 13) or all the QoS_{indivP} metrics (see *Overall* column in Table 13), whereas the same color of the adjacent uncertain situations indicates the same group, implying that the robustness difference across the 90 dispatchers under these uncertain situations are negligible. A lower count (y-axis) indicates a higher robustness under a specific uncertain situation. Each dot represents the average count across the 90 dispatchers on each sample, and the length of each line indicates twice of the standard deviation of each sample.

Conclusions for RQ2.2: **1)** The impact of the 15 uncertain situations on dispatchers' QoS_{indivP} robustness is also related to the templates, and more differences observed on *UpPeak* (23%) than on *LunchPeak* (12%); **2)** The 15 uncertain situations exhibit highly diverse impact on the QoS_{indivP} robustness across the 90 dispatchers, in terms of *TT*, during *UpPeak*; **3)** In general, interactions among uncertain factors *Capacity Factor*, *Loading Time* and *Unloading Time*, and interactions among all the four uncertain factors have relatively small impact on the QoS_{indivP} robustness, while uncertain factors *Mass* and *Capacity Factor* exhibit relatively large impact on the QoS_{indivP} robustness, across the 90 dispatchers.

4.2.3 RQ2.3: Rank QoS_{indivP} metrics. We further studied the degree to which each QoS_{indivP} metric is impacted by passengers' uncertainties by quantifying the robustness of all the 90 dispatchers in terms of each QoS_{indivP} metric with Eq. 25 on each template. The sorted QoS_{indivP} metrics for the 90 dispatchers on the *LunchPeak* and *UpPeak* are shown in Table 10. We can observe that the 90 dispatchers achieved the highest count in terms of *TT* for *LunchPeak*, whereas they have the lowest count in terms of *TT* for *UpPeak*. This might tell that the impact of passengers' uncertainties on a specific QoS_{indivP} robustness of the dispatchers is also related to the templates.

Table 10. QoS_{indivP} robustness of the 90 dispatchers - RQ2.3. For each QoS_{indivP} metric, there are, in total, 90 (dispatchers) × 15 (uncertain situations) × 10 (uncertain traffic profiles) = 13500 p-values. Each *Sorted Time List* cell tells the count calculated with Eq. 25. A lower count indicates a higher robustness.

Template	Sorted QoS_{indivP} Metrics			Total Count
LunchPeak	TT (2728)	WT (1999)	TD (1880)	6607
UpPeak	WT (4420)	TD (3479)	TT (3424)	11323



(a) Template specific ranking - *LunchPeak*. (b) Template specific ranking - *UpPeak*. (c) Overall ranking.

Fig. 12. The *SD-G* ranking of the three QoS_{indivP} metrics across the 90 dispatchers - RQ2.3. The x-axis shows the QoS_{indivP} metrics sorted according to their counts, whereas the same color of the adjacent QoS_{indivP} metrics indicates the same group (e.g., *TD* and *TT* in Fig. 12 (b) are in the same group), implying that the robustness difference in terms of these QoS_{indivP} metrics are negligible. A lower count (y-axis) indicates a higher robustness. Each dot represents the average count across the 90 dispatchers on each sample, and the length of each line indicates the twice of the standard deviation of each sample.

To further compare the robustness across the 90 dispatchers in terms of the three QoS_{indivP} metrics, we performed a ranking with Eq. 26 for each template. Results are shown in Fig. 12 (a) and (b), respectively. We can see that for *LunchPeak*, the three QoS_{indivP} metrics were categorized into one group, implying that the robustness differences across the 90 dispatchers in terms of the three QoS_{indivP} metrics are negligible. For *UpPeak*, *WT* was the only QoS_{indivP} metric categorized into the first group, which means that the 90 dispatchers achieved the worst robustness in terms of *WT*. One possible reason is that a majority of passenger activities during *UpPeak* are incoming, i.e., most people come to the office around 9:00 am; they take elevator from the entrance floor to their working floor, which might lead to the elevator frequently transit between the entrance floor and the top floor, and frequently stop to unload passengers working at different floors. As a result, the waiting time for the passengers increases. Consequently, the 90 dispatchers were relatively more sensitive to passengers' uncertainties in terms of *WT*. To compare the overall robustness across the 90 dispatchers in terms of the three QoS_{indivP} metrics on both templates, we conducted ranking again with Eq. 26 by combing the two templates, as shown in Fig. 12 (c). It is clear that the robustness of the 90 dispatchers when measured with the three different QoS_{indivP} metrics, have negligible differences, as they are ranked into one group.

Conclusions for RQ2.3: **1)** The impact of the passengers' uncertainties on the QoS_{indivP} robustness of the 90 dispatchers varies from template to template; **2)** The *WT* impacted the most by passengers' uncertainties on *UpPeak* across all the dispatchers; **3)** Overall, the robustness of the 90 dispatchers when measured with the three different QoS_{indivP} metrics have negligible differences.

4.3 Overall Discussions

We further looked at the results from the overall combined perspective of both the QoS_{allP} and QoS_{indivP} robustness. To do so, we summed the counts in terms of all the six QoS_{allP} metrics with Eq. 28 and the counts in terms of all the three QoS_{indivP} metrics with Eq. 30, on each template. Results are summarized in the *Total Count* column in Table 9 and Table 10, respectively. As we can see, the total counts on *UpPeak* are higher than that on *LunchPeak*, consistently for both the QoS_{allP} and QoS_{indivP} robustness. The main reason behind this could be that the *UpPeak* template is more demanding than the *LunchPeak*.

For the impact of the different uncertain situations on the robustness of all the 90 dispatchers, when we look at the overall ranking of the 15 uncertain situations in terms of the QoS_{allP} robustness (Fig. 8 (o)) and the QoS_{indivP} robustness (Fig. 11 (i)), we can observe that *usC*, *usM* and *usM-C* are the only three uncertain situations categorized into the first group, while *usC-L-U* was categorized into the last group, in both rankings. It means that uncertain factors *Capacity Factor*, *Mass*, and their interactions have more impact on the robustness of the dispatchers. In contrast, interactions among uncertain factors *Capacity Factor*, *Loading Time* and *Unloading Time* have relatively less impact on the robustness across all the 90 dispatchers.

When looking at specific dispatchers, as shown in Fig. 7 (c) and Fig. 10 (c), dispatcher version D02 was categorized into the last group, while D44 was categorized into the first group, in the overall rankings of both the QoS_{allP} and QoS_{indivP} robustness. It means that D02 has the best robustness and even performed more robust than the original dispatcher when dealing with passengers' uncertainties. We have confirmed with the domain experts in Orona that changes to the original dispatcher, in most cases, might have negative impacts on its performance; however, it might also be possible that a few changes may lead to better performance in certain situations. A reason for this could be that the dispatcher highly relies on parameters to be adaptable to different types of buildings. These parameters are optimized for each type of building based on the building needs. However our experiments used the original dispatcher with default parameter settings from Orona. Possibly, the parameters of D02 are optimized both for the building we have studied as well as for both traffic templates we have employed in our empirical evaluation. Conversely, D44 performed the least robust among all the dispatchers. In addition, one can also observe that the original dispatcher was categorized into the fifth group on the right in both overall rankings (Fig. 7 (c) and Fig. 10 (c)), which implies that the original dispatcher performed more robust than a majority of the 89 generated versions across the 15 uncertain situations.

We would also like to point it out that due to the current limitation of the simulator, we cannot control any temporal characteristics of the generation and interaction of uncertain factors at the simulation time. This is because the employed simulator, i.e., Elevate, requires previously generated traffic profiles before executing the simulation.

Finally, we summarize key observations and corresponding a set of recommendations from RQ1 and RQ2 in Table 11 for elevator designers.

Table 11. Summarized observations from RQ1-2, and recommendations for elevator practitioners.*

RQs	**Observations**	**Recommendations**
RQ1	Different metrics, i.e., *AWT*, *LWT*, *ATT*, *LTT*, *ATD*, and *LTD* impact the robustness differently under passengers' uncertainties. Overall, the dispatchers' robustness in terms of average time-related *QoS* metrics (i.e., *AWT*, *ATT* and *ATD*) are relatively less impacted than in terms of the longest time-related *QoS* metrics (i.e., *LWT*, *LTT* and *LTD*) by passengers' uncertainties.	Test and optimize elevators' robustness in terms of a specific *QoS* metric based on customer preferences. Otherwise, all the *QoS* metrics are recommended to be used for a comprehensive assessment, since we observed that the optimized *AWT* might lead to the increased *LTT*, suggesting that the optimization process may have to find the right balance of satisfying more than one metric.
RQ2	The 15 uncertain situations exhibit highly diverse impact on the robustness of the dispatchers in terms of *TT* (than *WT* and *TD*) on *UpPeak*.	Prioritize assessing a dispatcher's robustness in terms of *TT* when evaluating on *UpPeak*.
RQ1&2	The robustness of most dispatchers is affected by passengers' uncertainties. The dispatchers' robustness in the presence of passengers' uncertainties varies from template to template. Overall, *UpPeak* impact the dispatchers' robustness more than *LunchPeak*.	Assess the robustness of dispatchers in the presence of passengers' uncertainties for a number of templates given the time budget, and *UpPeak* should be prioritized for assessment.
	There is no correlation between the higher number of uncertain factors and the low robustness. Uncertain situations with more interacting uncertain factors do not necessarily lead to worse robustness.	All possible uncertain situations are recommended for robustness assessment, if there is a sufficient time available. Otherwise, start assessing the robustness with a single uncertain factor first then gradually assess with the interactions consisting of higher number of uncertain factors. This will give an indication of which uncertain factor(s) are affecting the robustness the most.
	Overall, uncertain factors *Capacity Factor*, *Mass*, and their interactions have more impact, than the other uncertain situations, on the dispatchers' robustness when evaluated on both templates.	Specific robustness optimization strategy for dealing with passengers' uncertain *Capacity Factor*, *Mass*, and their interactions should be investigated.

*Note that *RQ1* studies the robustness in terms of *QoS* provided to all the passengers as a whole, while *RQ2* focuses on the robustness in terms of *QoS* provided to each individual passenger; Row *RQ1&2* summarize the observations derived from both *RQ1* and *RQ2*, i.e., observations from both types of robustness.

5 GUIDELINES AND GENERALIZATION OF UNCERROBUA

In this section, we first present the guidelines of *UncerRobua* in Section 5.1, followed by its generalization to other software domains in Section 5.2.

5.1 Guidelines of *UncerRobua*

The methodology of conducting our empirical study is a systematic way to assess a dispatcher's robustness against passengers' uncertainties. This methodology is useful for guiding elevator designers to assess and investigate elevator dispatchers' robustness from various aspects, as the feedback from our industrial partner tells. We therefore derive the *UncerRobua* methodology for achieving the four objectives, as presented in Fig. 13.

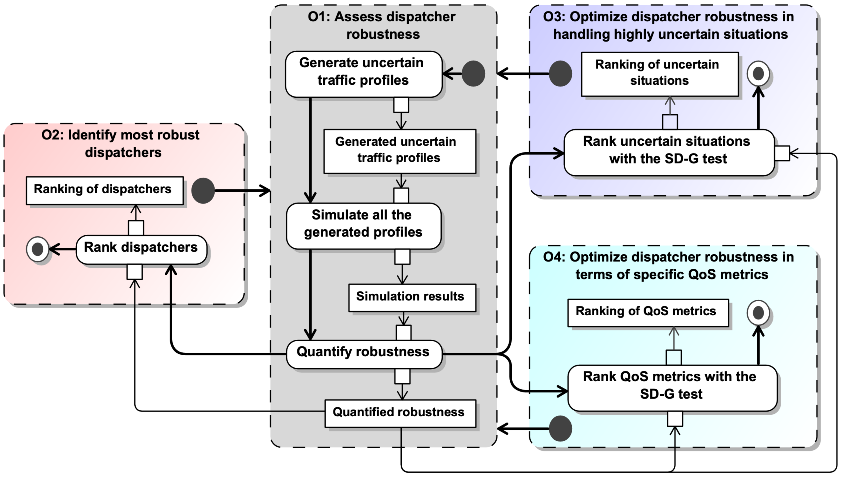

O1: Assess dispatcher robustness
Generate uncertain traffic profiles
Generated uncertain traffic profiles
Simulate all the generated profiles
Simulation results
Quantify robustness
Quantified robustness
O2: Identify most robust dispatchers
Ranking of dispatchers
Rank dispatchers
O3: Optimize dispatcher robustness in handling highly uncertain situations
Ranking of uncertain situations
Rank uncertain situations with the SD-G test
O4: Optimize dispatcher robustness in terms of specific QoS metrics
Ranking of QoS metrics
Rank QoS metrics with the SD-G test

Fig. 13. Guidelines for applying *UncerRobua*

Objective 1 (*O1*): Assessing dispatcher's robustness. An elevator dispatcher usually goes through a sequence of releases due to bug fixing, performance improvement, etc. Before each release, testing is performed to ensure the quality of the dispatcher. One important aspect is to test the robustness of the algorithm under various passengers' uncertainties. Such testing checks whether a new release has expected QoS_{allP} and QoS_{indivP} robustness as determined by domain experts. The essential activities of achieving *O1* are three steps: generating uncertain traffic profiles with different templates using Algorithm 1, simulate them, and quantify robustness (Section A.1), with the output being quantified robustness of the given dispatcher.

Objective 2 (*O2*): Identifying most robust dispatchers. Doing so can help elevator designers select the most robust dispatchers for a given building configuration. This is important because an elevator company, such as Orona, typically has various dispatchers for different building types (e.g., airports and hospitals). Moreover, comparing the robustness of different dispatchers is also useful to identify the weaknesses of the algorithms and therefore provide useful information for refining the algorithms to make them more robust. The first step is to generate uncertain traffic profiles, followed by simulating them with all the dispatchers that the elevator designer plans to investigate. Based on the simulation results, the robustness of each dispatcher is quantified, results of which are consequently used by the *SD-G* test to rank the robustness of the dispatchers in terms of all the QoS_{allP} metrics (with Eq. 9) and all the QoS_{indivP} metrics (with Eq. 19). Note that *O1* checks whether a dispatcher satisfies the expected robustness determined by domain experts, or whether the robustness of the dispatcher is degraded after, e.g., bug fixing; whilst *O2* ranks the robustness of all the dispatchers under investigation with the aim of identifying the most robust dispatcher(s).

Objective 3 (*O3*): Optimizing dispatcher robustness in handling highly uncertain situations. When designing elevator systems, in addition to satisfying all functional requirements, engineers need to empower elevator systems to elegantly deal with uncertainties from passengers or those caused by hardware failures such as delays in opening a door. To do so, elevator engineers need to systematically investigate various uncertain situations and their potential impact on the elevator systems. Based on the results of the investigation, they can then focus on uncertain situations that have a relatively large impact on the robustness of the systems. In addition to the essential activities of *O2*, to achieve *O3*, one also needs to rank the uncertain situations with the *SD-G* test. This resulted ranking of the uncertain situations provides elevator designers information about which uncertain situation(s) should be focused on.

Objective 4 (*O4*): Optimizing dispatcher robustness in terms of specific QoS_{allP} or QoS_{indivP} metric(s). Different QoS_{allP} and QoS_{indivP} metrics are usually affected differently by passengers' uncertainties. To compare the overall robustness in respect to each QoS_{allP}/QoS_{indivP} metric, the *SD-G* test can be performed to rank all the QoS_{allP}/QoS_{indivP} metrics across multiple executed results. The first three activities are the same as for *O1*, *O2* and *O3*. The *SD-G* test is, in the end, employed to rank all the QoS_{allP}/QoS_{indivP} metrics. The obtained ranking provides elevator designers the information on which QoS_{allP} or QoS_{indivP} metric(s) are impacted the most by passengers' uncertainties such that they can be the targets for optimization.

It is worth noting that it is essential to generate uncertain traffic profiles with values of uncertain factors within reasonable intervals without having the values deviating significantly from the baseline profile. Otherwise, generated uncertain traffic profiles would not reflect real situations, and consequently result in unreliable robustness assessment results. Nonetheless, in the real-world, uncertain situations could possibly result in values of an uncertain factor deviating very much from recommended values. Until now, we have not investigated how such situations could affect the *UncerRobua* methodology. Furthermore, we assessed the robustness in the SiL setting, and further experimentation is needed to see if *UncerRobua* is applicable in the HiL setting and could lead to different results as for the SiL setting.

5.2 Generalization of *UncerRobua*

Similar to elevator systems, most CPSs have the following characteristics: operating in uncertain environments; having directly (or indirectly) measurable *QoS*; and being required to handle uncertainties robustly. We therefore think that *UncerRobua* can be generalized to assess the robustness of other CPSs against uncertainties, and summarize the general process of applying *UncerRobua* as follows:

Generating uncertain profiles. To investigate the robustness of a system against uncertainties, uncertain factors that the system needs to handle during its operation should be firstly identified and specified with specific data types, e.g., *interval*, *distribution*, or *FuzzySet* [62]. With specified uncertain factors, uncertain profiles can be generated with *UncerRobua* to simulate uncertain situations. However, our current generator only handles uncertainties modeled with interval data types; therefore, other generators must be implemented depending on the data types. Note that various types of uncertain profiles can be generated based on a single uncertain factor or interactions, which has been focused on in many fields (e.g., [27, 64]), between multiple uncertain factors. Taking an example of Autonomous Driving System (ADS), uncertain factors can be the weather, pedestrians and surrounding vehicles, etc [29]. An uncertain situation can be caused by, e.g., only pedestrians, or, e.g., interactions between pedestrians and weather conditions. Correspondingly, uncertain profiles can be generated by introducing only pedestrians, or both pedestrians and foggy weather, for instance, to the environment configuration file.

Quantifying robustness. Executing generated uncertain profiles produces multiple outputs, and values of *QoS* metrics thus can be derived. For example, the *QoS* metrics of the ADS can be time to destination, trajectory offset and balancing of the car, etc. By comparing the values of the *QoS* metrics of the system operating under a specific uncertain situation with that of the baseline (without uncertainty) using, e.g., statistical tests, one can observe whether the *QoS* of the system is significantly degraded in the presence of uncertainty, which reflects the robustness of the system against a specific uncertain situation. Based on the observation, the robustness of the system in terms of a specific *QoS* metric, against a specific uncertain situation, thus can be quantified. Multiple uncertain situations (e.g., X) and *QoS* metrics (e.g., Y) correspond to multiple observations (e.g., $X \times Y$ in total). To systematically investigate and quantify systems' robustness against uncertainties, all the uncertain situations caused by all the captured uncertain factors (refer to Definitions 5 and 7 in the elevation domain), and comprehensive *QoS* metrics, are recommended to be studied.

Ranking with the SD-G test. To compare the overall robustness of, e.g., different software versions of a system, the *SD-G* test can be used to rank all the compared versions across all the uncertain situations based on the quantified robustness of each software version in terms of all the *QoS* metrics under each uncertain situation. To investigate the robustness of a specific software version under different uncertain situations, or in terms of different *QoS* metrics, with the final aim of prioritizing optimization scenarios, the *SD-G* test is recommended. For example, ranking the quantified robustness under each uncertain situation across multiple executions to prioritize uncertain situations that have a high impact on the robustness, or ranking the quantified robustness in terms of each *QoS* metric across all the uncertain situations to prioritize *QoS* metrics to be optimized. Note that ranking with *SD-G* test is generic so it is applicable to other contexts.

6 THREATS TO VALIDITY

In this section, we discuss threats to the validity of our empirical study.

6.1 Internal Validity

Threats to the internal validity is related to the internal factors [40]. For our empirical study, the first threat is the selection of parameter settings for the SiL simulation. To reduce this threat, we used an office building settings from the CIBSE Guide D [8]. Correspondingly, we used two CIBSE modern office templates from Elevate, which are based on multiple traffic surveys conducted by Peters Research Ltd. The two traffic templates model passengers' activities during *UpPeak* and *LunchPeak* in an office building, respectively. We did not tune the traffic flow parameters, e.g., *Arrival Floor*, as changing them will make the traffic flow of the traffic template provided by Elevate unrealistic. The second threat is that we only used one type of dispatchers. Orona, indeed, has various types of dispatchers, and we used the type that has been widely used in many elevators. In addition, the selection of the intervals for generating values of the uncertain factors (Section 3.1.5) might affect results of the empirical study, though we have no evidence of concluding it. Considering the intervals as independent variables requires a dedicated empirical study, which, unfortunately, we cannot afford currently.

6.2 Construct Validity

To minimize threats to the construct validity, first, we carefully selected *QoS* metrics. These metrics have been widely used in practise and are also comparable across all the 89 generated versions and the original algorithm. Second, to perform the ranking that considers the statistical difference among variables, we used the Cliff's Delta effect size. There exist other effect size measures such as

Cohen's d and Hedges' g; however, applying these measures require that samples are normally distributed [34], which is not the case for some of our data.

Another threat is that we generated uncertain values of the uncertain factors and controlled the average value of the generated uncertain values being the same as the recommended value in the standard [8] (e.g., 75 KG for *Mass*), to avoid deviating too much from the standard. There are other ways of generating uncertainties, e.g., using Poisson distributions [44], which we intend to investigate in the future work, if data are available.

6.3 Conclusion Validity

One threat to the conclusion validity is the number of uncertain traffic profiles we generated for each uncertain situation. More uncertain traffic profiles will bring more confidence in the results. To reduce this threat, for each uncertain situation, we generated 10 uncertain traffic profiles. Although 10 may not sound like a lot, it already resulted in a huge amount of simulations. In total, we performed $(15 \times 10 + 1) \times 2 \times (1 + 89) = 27180$ simulations, and each simulation simulated more than 1000 passengers. Another threat is the modeled passengers' uncertainties. Regarding this threat, we used intervals around the recommended values from the CIBSE Guide D [8] to generate values of the uncertain factors, which is close to passengers' uncertainties in real life and does not lead to unrealistic uncertainties. In the future, we plan to conduct more experiments with other traffic templates and more uncertain factors.

6.4 External Validity

A common threat to the external validity in empirical studies is lacking real case studies. To reduce this threat, we conducted the experiments with a real industrial elevator system from Orona. To generalize the results and hence mitigate the threat to the external validity, we generated a set of dispatcher versions by injecting small syntactic variations (check Section 3.1.2 for more details), which allowed us carrying out a more thorough empirical study with a larger dataset. Here, we want to acknowledge that, in an ideal situations, more real dispatchers could have been used. However, due to practical challenges, we had no access to them.

Another external validity threat is the generalization of our approach to other systems. As discussed in Section 5.2, some aspects of the approach, such as the ranking algorithm, can be used in other contexts without modifications, whereas certain aspects such as uncertainty generation need to be extended depending on the system. In the future, we plan to build generic and configurable interfaces to ease its use in other contexts further.

7 RELATED WORK

We present the related work from three aspects: testing under uncertainty (Section 7.1), testing of the elevator systems (Section 7.2) and elevator uncertainty (Section 7.3).

7.1 Testing under Uncertainty

Uncertainty is a growing concern in complex CPSs [11]. Accordingly, researchers have proposed approaches for testing CPSs under uncertainty. For example, Zhang et al. proposed *UncerTest* [61], a model-based testing tool for generating uncertainty-aware test cases to test CPS under environmental uncertainties. Camilli et al. [9] proposed an online model-based testing approach that dynamically generates test cases with the uncertainty sampling policy, which takes into account the uncertain parameters of the Markov Decision Process (MDP) model built for the investigated Tele Assistance System. Ali et al. [4] systematically investigated uncertainty-wise testing, in the context of CPSs, from various aspects, e.g., uncertainty-wise model-based testing and uncertainty-wise multi-objective test optimization, and presented a vision toward the research directions of uncertainty testing. To test CPS Simulink models, Menghi et al. [36] proposed *SOCRaTeS* for generating test oracles, where white noise signals were employed to simulate the noise in the inputs especially caused by sensor readings, and uncertain parametric values were used to model unknown hardware choices that lead to undetermined model parameters. Walkinshaw et al. [50] proposed a black-box testing framework, which infers behavioral models of the SUT using Genetic Programming (GP) to select the most uncertain test cases for the inferred model for execution, thus decreasing uncertainty about the correctness of a software system. Ul Haq et al. [49] proposed *SAMOTA* for test suite generation, which considers uncertainty of surrogate models' predictions. When comparing the above works with *UncerRobua*, none of the above work assesses the robustness of CPS against uncertainties. Instead most of them focus on the generation ([9, 36, 49, 61]) and optimization ([4, 50]) of uncertainty-related test cases. Nonetheless, *UncerRobua* can be used to generate uncertainty-aware robustness tests for CPSs uncertainty, which is one of our future works.

Ma et al. [32] proposed two algorithms to test the self-healing behaviors of CPSs: Fragility-Oriented Testing (FOT) for faults detection and Uncertainty Policy Optimization (UPO) for uncertainty generation. FOT utilizes the fragility to learn the optimal strategy for operation invocation, once a sequence of operation invocations is selected, UPO is used to introduce uncertainty, i.e., learn the optimal uncertainty values of the state variables to make the CPS behave under uncertainty, based on the fragility obtained from test executions. To test the self-healing behaviors of CPSs under environmental uncertainties, Ma et al. [30] proposed a model-based approach consisting of a modeling framework of self-healing CPSs (MoSH) and a test model executor (TM-Executor). Based on their previous works [32] and [30], Ma et al. [31] performed an empirical study, using reinforcement learning algorithms to test self-healing CPSs under uncertainty. This work provides an evidence about the cost-effectiveness of different reinforcement learning algorithms for testing self-healing behaviors. To compare with the work of Ma et al. [30–32], we, however, take a new uncertainty generation approach, which studies all possible interactions between different uncertain factors instead of merging all uncertainties together, and particularly focus on passenger uncertainties in the context of an industrial elevator system. It is worth noting that our approach also ranks all the uncertain situations based on their impact on the system, which can be used to prioritize uncertainty testing scenarios.

In summary, uncertainty test case generation and optimization are still the main focus of the literature (e.g., [4, 9, 36, 49, 50, 61]). Furthermore, model-based testing (e.g., [9, 10, 12, 61, 63]) and search-based algorithms (e.g., [4, 28, 42, 49, 59]) are widely used in uncertainty testing of CPSs. These works have the risk of generating extremely uncertain and possibly unrealistic situations, which are highly likely to affect the robustness of the system. In contrast, our work *UncerRobua* systematically assess the robustness of CPSs against uncertainties that occur in realistic situations. In our opinion, both lines of research and techniques have different aims but are complementary to each other.

7.2 Testing Elevator Systems

Testing elevator systems has been investigated with various techniques, e.g., metamorphic testing [6], machine learning [5, 56], and model-based development [37]. However, most of them focus on testing functionalities, e.g., braking [20, 33], and doors [43] of elevator systems. For instance, Peng et al. [38] proposed a dynamic braking torque test method to test both the static and emergency braking torque. Wu et al. [54] proposed a thermal elastoplastic contact model. This model treats the frictional sliding during braking as the interfacial failure of the contact material, for studying the frictional braking of elevator safety gear. In terms of elevator brake failure, Wang et al. [55] developed an elevator braking system evaluation model using fishbone diagrams for analyzing the problems of elevator braking systems.

A few works focus on testing the performance of elevator systems. Chen et al. [13] investigated the cooling performance of the air conditioner of elevator cars and optimized the cooling capacity by adjusting the input current. They increased the current from 1.5A to 9.9A and observed that the maximum air-cooling capacity can be obtained at 9.4A. Al-Sharif et al. [3] proposed a paradigm with three components for testing the performance of elevator group control algorithms using idealized benchmarks, random scenario testing, and gradually making scenarios more realistic. Wang et al. [51] tested the seismic performance of a functional traction elevator in a reinforced concrete building under earthquake. The designed building can shake in east-west direction, which takes advantage of the ground motions developed for a site in Southern California.

In summary, existing works mainly tested elevator systems' functionalities, and a few works focused on the performance testing. However, there is no relevant research on the robustness assessment of elevator systems in dealing with passengers' uncertainties. Our work is a novel method for systematically assessing the robustness of elevator dispatchers against passengers' uncertainties.

7.3 Elevator Uncertainty

Uncertainties in elevator systems have been increasingly investigated. For example, in [2], Monte Carlo simulations were used to evaluate passengers' average traveling time during up peak with environmental uncertainties, e.g., floor populations and heights, and multiple entrances. In addition, Fuzzy theory has also been used for various purposes, e.g., measuring passengers' satisfaction [57], minimizing waiting time [18], and optimizing energy consumption [17], in the presence of uncertainties, i.e., passengers' uncertain waiting time [57] and calls [17, 18]) in elevator systems. To reduce elevator systems' energy consumption in the context of uncertain traffic patterns, Zhang et al. [60] proposed an energy-saving strategy for elevator dispatching. The strategy selects one of their developed energy-saving dispatching models, i.e., up-peak, down-peak, up/down-mixed, and night dispatching models, based on real time traffic pattern. Then the dispatching optimization solutions can be obtained by solving the models using Linear Interactive and General Optimizer software.

In a recent study on elevators' brake systems, Wolszczak et al. [53] proposed an optimization framework to improve the efficiency of the elevator's brake system under braking force uncertainties, which are caused by the brake cam angle and the spring reaction force. Passengers' uncertainties have also been investigated recently. For example, Sorsa et al. [44] investigated elevator dispatching problem, in which passengers' arrival uncertainty are modeled with Poisson and geometric Poisson distributions. They formulated elevator dispatching problem as a stochastic optimal control problem, for which they also designed a robust optimization approach by considering multiple parametric values of all possible stops of the elevator across all the floors. Evaluation was performed by comparing estimated values (e.g., passenger's demand reflected as the number of waiting passengers behind a call, and waiting time) with actual values generated with simulations. Results show that the geometric Poisson process performs better than the Poisson process in estimating uncertainty. Different from [44], we didn't model passengers' arrival uncertainty with distributions by ourselves. Instead, we adopted template specific arrival data (i.e., *Arrival Time* and *Arrival Floor*) generated by Elevate, which are based on multiple traffic surveys conducted on operational elevators in real buildings, as also evidenced in Fig. 3 and Fig. 4.

Software uncertainty has also been considered recently. For instance, to reduce passengers' waiting time, Bapin et al. [7] proposed an optimization algorithm for elevator dispatching using the information obtained from a real-time surveillance video processed by an image processing system, which estimates the number of passengers waiting for an elevator in hallway and predicts passengers' movement directions. The proposed method takes into account the uncertainty due to the possible errors caused by the image recognition software, e.g., misidentifying luggage as passengers. This kind of uncertainty is solved utilizing the Bayesian network [19].

In summary, existing works mainly focused on designing optimization methods for elevator systems under uncertainties, from different aspects such as energy consumption, braking efficiency, and dispatching problems. In contrast, our work focuses on assessing the robustness of industrial elevator dispatchers against passengers' uncertainties from various aspects. Furthermore, the uncertainties we studied cover more uncertain factors of passengers and we also investigated all the possible interactions between multiple uncertain factors, which has not yet been investigated in existing works.

8 CONCLUSION AND FUTURE WORK

Elevator systems transport passengers in the presence of various uncertainties, e.g., software, hardware, and environment uncertainties. These uncertainties affect elevator systems in different ways, such as decreasing dispatching efficiency, worsening systems' robustness, and even threatening systems' security [21]. Among all the uncertainties, passengers' uncertainties are the most common and exist throughout the operating cycle of elevator systems.

To study the impact of passengers' uncertainties on the robustness of elevator dispatchers (i.e., software), we performed a comprehensive industrial case study. We investigated such uncertainties in the form of various uncertain situations caused by passengers' uncertain factors and their possible interactions. With an industrial elevator dispatcher from Orona, we conducted an empirical evaluation. During the assessment, both QoS_{allP} robustness and QoS_{indivP} robustness were investigated, each of which is further analyzed from the following aspects: 1) assessing each dispatcher's robustness against passengers' uncertainties in terms of QoS_{allP}/QoS_{indivP} metrics; 2) investigating the impact of different uncertain situations on the QoS_{allP}/QoS_{indivP} robustness of the dispatchers; and 3) investigating which QoS_{allP}/QoS_{indivP} metric(s) is impacted the most by passenger uncertainties. To perform ranking, which takes into account statistical differences, for various comparison purposes, e.g., ranking the uncertain situations, or the dispatchers, we developed the *SD-G* test. Based on the experience learned in the empirical study, we finally derived *UncerRobua*, a comprehensive Uncertainty-aware Robustness assessment methodology along with a set of guidelines for elevator designers to systematically investigate the robustness of dispatchers. To facilitate the application of *UncerRobua* by practitioners in other software domains, we also discussed the generalization of *UncerRobua*.

In the future, we plan to 1) Investigate more uncertain factors as well as other types of uncertainties such as hardware uncertainty; 2) Build a conceptual model covering all types of uncertainty in elevator systems to provide guidance for elevator designers to systemically investigate elevator systems' uncertainties; 3) Classify various uncertain situations into different levels/categories based on the degree of their impact on dispatchers, which can guide elevator designers in designing uncertainty handling solutions, e.g., optimization strategies specific to certain types of uncertainties; 4) Investigate more traffic templates, e.g., *DownPeak*; 5) Study different uncertainty generation methods, e.g. model passengers' uncertain attributes with distributions; 6) Investigate advanced uncertainty sampling methods such as with Monte Carlo simulation.

ACKNOWLEDGMENTS

This work is supported by the Adeptness project funded by the European Union's Horizon 2020 programme under Grant Agreement no. 871319. Aitor Arrieta is part of the Software and Systems Engineering research group of Mondragon Unibertsitatea (IT1519-22), supported by the Department of Education, Universities and Research of the Basque Country. We thank Orona for supporting the empirical study. The results and discussions do not, in any way, represent the quality of Orona's elevators. We also thank Jon Ayerdi Cantalejo for helping us to run simulation and Christoph Laaber for constructive discussions on statistical analyses.

REFERENCES

[1] Hiralal Agrawal, Richard DeMillo, R_ Hathaway, William Hsu, Wynne Hsu, Edward W Krauser, Rhonda J Martin, Aditya P Mathur, and Eugene Spafford. 1989. *Design of mutant operators for the C programming language*. Technical Report. Citeseer.

[2] Lutfi Al-Sharif, OF Abdel Aal, and AM Abu Alqumsan. 2013. Evaluating the elevator passenger average travelling time under incoming traffic conditions using analytical formulae and the Monte Carlo method. *Elevator World* 61, 6 (2013), 110–123.

[3] Lutfi Al-Sharif, Zaid Jaber, Jamal Hamdan, and Anas Riyal. 2016. Evaluating the performance of elevator group control algorithms using a three-element new paradigm. *Building services engineering research and technology* 37, 5 (2016), 597–613. https://doi.org/10.1177/0143624416652182

[4] Shaukat Ali, Hong Lu, Shuai Wang, Tao Yue, and Man Zhang. 2017. Uncertainty-wise testing of cyber-physical systems. In *Advances in Computers*. Vol. 107. Elsevier, 23–94. https://doi.org/10.1016/bs.adcom.2017.06.001

[5] Aitor Arrieta, Jon Ayerdi, Miren Illarramendi, Aitor Agirre, Goiuria Sagardui, and Maite Arratibel. 2021. Using Machine Learning to Build Test Oracles: an Industrial Case Study on Elevators Dispatching Algorithms. In *2021 IEEE/ACM International Conference on Automation of Software Test (AST)*. 30–39. https://doi.org/10.1109/AST52587.2021.00012

[6] Jon Ayerdi, Sergio Segura, Aitor Arrieta, Goiuria Sagardui, and Maite Arratibel. 2020. QoS-aware Metamorphic Testing: An Elevation Case Study. In *2020 IEEE 31st International Symposium on Software Reliability Engineering (ISSRE)*. 104–114. https://doi.org/10.1109/ISSRE5003.2020.00019

[7] Yerzhigit Bapin, Kanat Alimanov, and Vasilios Zarikas. 2020. Camera-driven probabilistic algorithm for multi-elevator systems. *Energies* 13, 23 (2020), 6161. https://doi.org/10.3390/en13236161

[8] Gina Barney. 2010. *Transportation systems in buildings : CIBSE Guide D: 2010*. London: Chartered Institution of Building Services Engineers.

[9] Matteo Camilli, Carlo Bellettini, Angelo Gargantini, and Patrizia Scandurra. 2018. Online Model-Based Testing under Uncertainty. In *2018 IEEE 29th International Symposium on Software Reliability Engineering (ISSRE)*. 36–46. https://doi.org/10.1109/ISSRE.2018.00015

[10] Matteo Camilli, Angelo Gargantini, Patrizia Scandurra, and Catia Trubiani. 2021. Uncertainty-aware Exploration in Model-based Testing. In *2021 14th IEEE Conference on Software Testing, Verification and Validation (ICST)*. 71–81. https://doi.org/10.1109/ICST49551.2021.00019

[11] Ferhat Ozgur Catak, Tao Yue, and Shaukat Ali. 2022. Uncertainty-aware prediction validator in deep learning models for cyber-physical system data. *ACM Transactions on Software Engineering and Methodology (TOSEM)* 31, 4 (2022), 1–31.

[12] Amrita Chatterjee and Hassan Reza. 2020. Toward Modeling and Verification of Uncertainty in Cyber-Physical Systems. In *2020 IEEE International Conference on Electro Information Technology (EIT)*. 568–576. https://doi.org/10.1109/EIT48999.2020.9208273

[13] Chengdai Chen, Lingbo Mao, Tao Lin, Teng Tu, Longqian Zhu, and Changhong Wang. 2020. Performance testing and optimization of a thermoelectric elevator car air conditioner. *Case Studies in Thermal Engineering* 19 (2020), 100616. https://doi.org/10.1016/j.csite.2020.100616

[14] Norman Cliff. 1993. Dominance statistics: Ordinal analyses to answer ordinal questions. *Psychological bulletin* 114, 3 (1993), 494.

[15] Jeroen Leonard De Jong. 2012. Heuristics in dynamic scheduling: a practical framework with a case study in elevator dispatching. (2012).

[16] Ning Ding, Tao Chen, and Hui Zhang. 2017. Experimental study of elevator loading and unloading time during evacuation in high-rise buildings. *Fire technology* 53, 1 (2017), 29–42.

[17] Joaquín Fernández, Pablo Cortés, José Guadix, and Jesús Muñuzuri. 2013. Dynamic fuzzy logic elevator group control system for energy optimization. *International Journal of Information Technology & Decision Making* 12, 03 (2013), 591–617.

[18] J Fernandez, Pablo Cortés, Jesús Muñuzuri, and José Guadix. 2013. Dynamic fuzzy logic elevator group control system with relative waiting time consideration. *IEEE transactions on industrial electronics* 61, 9 (2013), 4912–4919.

[19] Nir Friedman, Dan Geiger, and Moises Goldszmidt. 1997. Bayesian network classifiers. *Machine learning* 29, 2 (1997), 131–163.

[20] Dimitrios Giagopoulos, Iraklis Chatziparasidis, and Nickolas S Sapidis. 2018. Dynamic and structural integrity analysis of a complete elevator system through a Mixed Computational-Experimental Finite Element Methodology. *Engineering Structures* 160 (2018), 473–487.

[21] Liping Han, Tao Yue, Shaukat Ali, Aitor Arrieta, and Maite Arratibel. 2022. Are Elevator Software Robust against Uncertainties? Results and Experiences from an Industrial Case Study. In *Proceedings of the 30th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*.

[22] Steffen Herbold. 2017. Comments on ScottKnottESD in response to" An empirical comparison of model validation techniques for defect prediction models". *IEEE Transactions on Software Engineering* 43, 11 (2017), 1091–1094.

[23] M Hermanussen, H Danker-Hopfe, and GW Weber. 2001. Body weight and the shape of the natural distribution of weight, in very large samples of German, Austrian and Norwegian conscripts. *International journal of obesity* 25, 10 (2001), 1550–1553.

[24] Jafferi Jamaludin, Nasrudin Abd Rahim, and Wooi Ping Hew. 2009. Development of a self-tuning fuzzy logic controller for intelligent control of elevator systems. *Engineering Applications of Artificial Intelligence* 22, 8 (2009), 1167–1178.

[25] Tuuli Kauppala et al. 2021. Developmental trajectories of height, weight and BMI across childhood: Bayesian hierarchical modeling of longitudinal data. (2021).

[26] Barbara Eva Kirunda, Lars Thore Fadnes, Henry Wamani, Jan Van den Broeck, and Thorkild Tylleskär. 2015. Population-based survey of overweight and obesity and the associated factors in peri-urban and rural Eastern Uganda. *BMC Public Health* 15, 1 (2015), 1–11.

[27] Melanie E Kreye. 2018. Interactions between perceived uncertainty types in service dyads. *Industrial Marketing Management* 75 (2018), 90–99.

[28] Jaekwon Lee, Seung Yeob Shin, Shiva Nejati, Lionel C Briand, and Yago Isasi Parache. 2020. Schedulability Analysis of Real-Time Systems with Uncertain Worst-Case Execution Times. *arXiv preprint arXiv:2007.10490* (2020).

[29] Chengjie Lu, Yize Shi, Huihui Zhang, Man Zhang, Tiexin Wang, Tao Yue, and Shaukat Ali. 2022. Learning Configurations of Operating Environment of Autonomous Vehicles to Maximize their Collisions. *IEEE Transactions on Software Engineering* (2022).

[30] Tao Ma, Shaukat Ali, and Tao Yue. 2019. Modeling foundations for executable model-based testing of self-healing cyber-physical systems. *Software & Systems Modeling* 18, 5 (2019), 2843–2873.

[31] Tao Ma, Shaukat Ali, and Tao Yue. 2021. Testing self-healing cyber-physical systems under uncertainty with reinforcement learning: an empirical study. *Empirical Software Engineering* 26, 3 (2021), 1–54.

[32] Tao Ma, Shaukat Ali, Tao Yue, and Maged Elaasar. 2019. Testing self-healing cyber-physical systems under uncertainty: a fragility-oriented approach. *Software Quality Journal* 27, 2 (2019), 615–649.

[33] Xiaolong Ma, Gen Pan, Peng Zhang, Qing Xu, Xi Shi, Zeliang Xiao, and Yunting Han. 2021. Experimental evaluation of braking pad materials used for high-speed elevator. *Wear* (2021), 203872. https://doi.org/10.1016/j.wear.2021.203872

[34] Philomena Marfo and GA Okyere. 2019. The accuracy of effect-size estimates under normals and contaminated normals in meta-analysis. *Heliyon* 5, 6 (2019), e01838.

[35] TAK Mathews and Raghavan Nalini. 2007. Vertical Transportation Configuration–Design Approach and Traffic Analysis Theory. *Journal of the Indian Institute of Engineers* 88 (2007).

[36] Claudio Menghi, Shiva Nejati, Khouloud Gaaloul, and Lionel C Briand. 2019. Generating automated and online test oracles for simulink models with continuous and uncertain behaviors. In *Proceedings of the 2019 27th acm joint meeting on european software engineering conference and symposium on the foundations of software engineering*. 27–38.

[37] Carlos Fernando Nicolas, Iban Ayestaran, Imanol Martinez, and Patricia Franco. 2016. Model-Based Development of an FPGA Encoder Simulator for Real-Time Testing of Elevator Controllers. In *2016 IEEE 19th International Symposium on Real-Time Distributed Computing (ISORC)*. 53–60. https://doi.org/10.1109/ISORC.2016.17

[38] Qifeng Peng, Zhongxing Li, Hong Yuan, Guojian Huang, Shanqing Li, and Xueli Sun. 2018. A model-based unloaded test method for analysis of braking capacity of elevator brake. *Advances in Materials Science and Engineering* 2018 (2018). https://doi.org/10.1155/2018/8047490

[39] Jeanine Romano, Jeffrey D Kromrey, Jesse Coraggio, and Jeff Skowronek. 2006. Should we really be using t-test and cohen'sd for evaluating group differences on the nsse and other surveys. In *Annual meeting of the Florida association of institutional research*.

[40] Per Runeson, Martin Host, Austen Rainer, and Bjorn Regnell. 2012. *Case study research in software engineering: Guidelines and examples*. John Wiley & Sons.

[41] Andrew Jhon Scott and M Knott. 1974. A cluster analysis method for grouping means in the analysis of variance. *Biometrics* (1974), 507–512.

[42] Seung Yeob Shin, Shiva Nejati, Mehrdad Sabetzadeh, Lionel C Briand, and Frank Zimmer. 2018. Test case prioritization for acceptance testing of cyber physical systems: a multi-objective search-based approach. In *Proceedings of the 27th acm sigsoft international symposium on software testing and analysis*. 49–60. https://doi.org/10.1145/3213846.3213852

[43] Isaac Skog, Ioannis Karagiannis, Anders Betts Bergsten, Jonas Härdén, Lars Gustafsson, and Peter Händel. 2017. A Smart Sensor Node for the Internet-of-Elevators—Non-Invasive Condition and Fault Monitoring. *IEEE Sensors Journal* 17, 16 (2017), 5198–5208. https://doi.org/10.1109/JSEN.2017.2719630

[44] Janne Sorsa, Harri Ehtamo, Juha-Matti Kuusinen, Mirko Ruokokoski, and Marja-Liisa Siikonen. 2018. Modeling uncertain passenger arrivals in the elevator dispatching problem with destination control. *Optimization Letters* 12, 1 (2018), 171–185.

[45] Chakkrit Tantithamthavorn. 2019. Non-parametric Scott-Knott ESD. https://github.com/klainfo/ScottKnottESD

[46] Chakkrit Tantithamthavorn, Shane McIntosh, Ahmed E Hassan, and Kenichi Matsumoto. 2016. An empirical comparison of model validation techniques for defect prediction models. *IEEE Transactions on Software Engineering* 43, 1 (2016), 1–18.

[47] Chakkrit Tantithamthavorn, Shane McIntosh, Ahmed E Hassan, and Kenichi Matsumoto. 2018. The impact of automated parameter optimization on defect prediction models. *IEEE Transactions on Software Engineering* 45, 7 (2018), 683–711.

[48] Toni Tukia, Semen Uimonen, Marja-Liisa Siikonen, Claudio Donghi, and Matti Lehtonen. 2018. High-resolution modeling of elevator power consumption. *Journal of Building Engineering* 18 (2018), 210–219.

[49] Fitash Ul Haq, Donghwan Shin, and Lionel Briand. 2022. Efficient Online Testing for DNN-Enabled Systems using Surrogate-Assisted and Many-Objective Optimization. In *Proceedings of the 44th International Conference on Software Engineering (ICSE'22)*. ACM.

[50] Neil Walkinshaw and Gordon Fraser. 2017. Uncertainty-Driven Black-Box Test Data Generation. In *2017 IEEE International Conference on Software Testing, Verification and Validation (ICST)*. 253–263. https://doi.org/10.1109/ICST.2017.30

[51] Xiang Wang, Tara C Hutchinson, Rodrigo Astroza, Joel P Conte, José I Restrepo, Matthew S Hoehler, and Waldir Ribeiro. 2017. Shake table testing of an elevator system in a full-scale five-story building. *Earthquake engineering & structural dynamics* 46, 3 (2017), 391–407.

[52] Claes Wohlin, Per Runeson, Martin Höst, Magnus C Ohlsson, Björn Regnell, and Anders Wesslén. 2012. *Experimentation in software engineering*. Springer Science & Business Media.

[53] Piotr Wolszczak, Pawel Lonkwic, Americo Cunha, Grzegorz Litak, and Szymon Molski. 2019. Robust optimization and uncertainty quantification in the nonlinear mechanics of an elevator brake system. *Meccanica* 54, 7 (2019), 1057–1069.

[54] Aizhong Wu, Xi Shi, Lin Weng, and Dingyu Hu. 2020. Thermo-mechanical modeling and transient analysis of frictional braking of elevator safety gear. *Journal of Thermal Stresses* 43, 12 (2020), 1467–1486. https://doi.org/10.1080/01495739.2020.1820921

[55] Wang Xiaolun and Xupeng Zhang. 2020. Research on Elevator Braking Failure Assessment Model Based on Fishbone Diagram and AHP. In *2020 International Conference on Artificial Intelligence and Computer Engineering (ICAICE)*. 260–263. https://doi.org/10.1109/ICAICE51518.2020.00056

[56] Qinghua Xu, Shaukat Ali, Tao Yue, and Maite Arratibel. 2022. Uncertainty- Aware Transfer Learning to Evolve Digital Twins for Industrial Elevators. In *Proceedings of the 30th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*.

[57] Dong Mei Yan and Li Liu. 2014. Research on elevator group control uncertainty. In *Applied Mechanics and Materials*, Vol. 602. Trans Tech Publ, 874–877.

[58] Olivia Hyunjung Yoo and Jiyoung Park. 2013. The Elevator-Integrated Delivery System for High-Rise Residential Buildings. *Journal of Asian Architecture and Building Engineering* 12, 1 (2013), 149–156.

[59] Huihui Zhang, Man Zhang, Tao Yue, Shaukat Ali, and Yan Li. 2020. Uncertainty-wise requirements prioritization with search. *ACM Transactions on Software Engineering and Methodology (TOSEM)* 30, 1 (2020), 1–54.

[60] Jinglong Zhang and Qun Zong. 2014. Energy-saving-oriented group-elevator dispatching strategy for multi-traffic patterns. *Building Services Engineering Research and Technology* 35, 5 (2014), 543–568. https://doi.org/10.1177/0143624414526723

[61] Man Zhang, Shaukat Ali, and Tao Yue. 2019. Uncertainty-wise test case generation and minimization for cyber-physical systems. *Journal of Systems and Software* 153 (2019), 1–21.

[62] Man Zhang, Shaukat Ali, Tao Yue, Roland Norgren, and Oscar Okariz. 2019. Uncertainty-wise cyber-physical system test modeling. *Software & Systems Modeling* 18, 2 (2019), 1379–1418.

[63] Man Zhang, Tao Yue, Shaukat Ali, Bran Selic, Oscar Okariz, Roland Norgre, and Karmele Intxausti. 2018. Specifying uncertainty in use case models. *Journal of Systems and Software* 144 (2018), 573–603.

[64] Shuai Zhou, Yimin Wang, Ziyan Li, Jianxia Chang, and Aijun Guo. 2021. Quantifying the uncertainty interaction between the model input and structure on hydrological processes. *Water Resources Management* 35, 12 (2021), 3915–3935.

A APPENDIX – DETAILED METRICS

A.1 Robustness quantification

A simulation of an uncertain traffic profile $P_{us_o,k}$ with a dispatcher compiled in Elevate produces values for the *Time List*[10] (see the example in Table 2) and QoS_{allP} metrics (see the example in Table 3). These values are represented as $TL_{us_o,k} = \{TL_{s,us_o,k} | s = 1, 2, 3\}$ and $Q_{us_o,k} = \{q_{l,us_o,k} | l = 1, ..., 6\}$, respectively, where $TL_{s,us_o,k}$ (see Definition 3) and $q_{l,us_o,k}$ are the sth *Time List* value and lth QoS_{allP} value for the kth uncertain traffic profile of the oth uncertain situation, respectively. For instance, assume the simulation results shown in Table 2 and Table 3 are generated by executing the 3rd uncertain traffic profile of the first uncertain situation us_1. Correspondingly, the value of the second QoS_{allP} can be represented as $q_{2,us_1,3}$ (i.e., the LWT value in Table 3), and a set of values for the second *Time List* can be represented as $TL_{2,us_1,3}$ (i.e., the TT column in Table 2). After executing simulations with all the generated uncertain traffic profiles of all the uncertain situations, we obtained our dataset. We then performed corresponding analyses on the dataset to answer the RQs (Section 3.2) with dedicated statistical tests.

To study the robustness of a dispatcher against passengers' uncertainties, in terms of each QoS_{allP} metric, we used the one-sample Wilcoxon signed rank test (one tailed, at the significance level of 0.05) to compare the NR values of one specific QoS_{allP} metric obtained for the NR uncertain traffic profiles with one value produced by the baseline profile (with fixed values for all the passengers' uncertain factors, see Section 3.1.4), which results the below p-value:

$$p^{qosa}_{l,us_o} = OneSampleWilcoxonTest\left(\left((q_1, q_2, ..., q_{NR})_{us_o}\right)_l, q^b_l\right) \tag{3}$$

where, p^{qosa}_{l,us_o} is the p-value under the oth uncertain situation us_o in terms of the lth QoS_{allP} metric; $\left((q_1, q_2, ..., q_{NR})_{us_o}\right)_l$ is an array composed of the values of the lth QoS_{allP} metric of the NR uncertain traffic profiles under uncertain situation us_o; q^b_l is the value of the lth QoS_{allP} metric of the baseline profile. For example, to study the robustness of a dispatcher against the second uncertain situation us_2 in terms of the first QoS_{allP} – AWT with 10 uncertain traffic profiles, the 10 produced AWT values ($\left((q_1, q_2, ..., q_{10})_{us_2}\right)_1$) are compared with the one (q^b_1) of the baseline profile with Eq. 3 to obtain p-value p^{qosa}_{1,us_2}. A resulting p-value less than 0.05 means that the uncertain situation significantly worsens the QoS_{allP} attribute significantly, i.e., the dispatcher performed less robust under a specific uncertain situation in terms of a specific QoS_{allP} metric. Taking the example above, $p^{qosa}_{1,us_2} < 0.05$ indicates that the 10 produced AWT values under uncertain situation us_2 are significantly greater than the one produced by the baseline profile.

Similarly, the comparison of the sth *Time List* generated by the kth uncertain traffic profile and the baseline profile is performed with the paired Wilcoxon signed rank test (one tailed, at a significance level of 0.05), as shown below:

$$p^{tl}_{s,us_o,k} = PairedWilcoxonTest\left(TL_{s,us_o,k}, TL^b_s\right) \tag{4}$$

where $TL_{s,us_o,k}$ is the sth *Time List* (with s being 1, 2 or 3, respectively, representing *Time List*s for WT, TT or TL (Section 2)) generated by the kth uncertain traffic profile under uncertain situation us_o; TL^b_s is the sth *Time List* of the baseline profile; $p^{tl}_{s,us_o,k}$ is the p-value corresponding to the kth uncertain traffic profile of the uncertain situation us_o in terms of the sth *Time List*. A resulting p-value less than 0.05 indicates that passengers' uncertainties have a significant impact on the QoS_{indivP}, which implies a relatively poor robustness of the dispatcher when dealing with passengers' uncertainties. For instance, the comparison between the first *Time List* (i.e., *WT Time List*) generated by the

[10]Note that Elevate only generates WT and TT, from which TD is calculated.

baseline profile (TL_1^b) and the *WT Time List* produced by executing the 3rd uncertain traffic profile corresponding to uncertain situation us_2 ($TL_{1,us_2,3}$) may result in $p^{tl}_{1,us_2,3} < 0.05$ with Eq. 4, which implies a relatively poor robustness of the dispatcher in terms of WT under uncertain situation us_2.

Based on generated p-values with Eq. 3 and Eq. 4, we then quantify the QoS_{allP} and QoS_{indivP} robustness of a dispatcher by counting the number of times that p-value<0.05 under each uncertain situation. The robustness of a dispatcher under uncertain situation us_o in terms of the lth QoS_{allP} and the sth QoS_{indivP} metrics can be quantified with Eq. 5 and Eq. 6, respectively:

$$Robustness^{QoSa}_{l,us_o} = Count\left(p^{qosa}_{l,us_o} < 0.05\right) \tag{5}$$

$$Robustness^{QoSi}_{s,us_o} = Count\left(p^{tl}_{s,us_o,k} < 0.05, \quad k = 1, \cdots, NR\right) \tag{6}$$

Where, the QoS_{allP} robustness measurements obtained with Eq. 5 can only be 0 or 1, with 1 indicating relatively poorer robustness, because a QoS_{allP} metric (e.g., *AWT*) measures the quality of service provided to all the passengers as a whole and hence one value is returned for each QoS_{allP} metric, after executing each traffic profile with Elevate. For QoS_{indivP} robustness, a quantification result can take any value from 0 to NR (the total number of the uncertain traffic profiles) with a higher value indicating a poorer robustness of the dispatcher against passengers' uncertainties. This is because, a *Time List* of each traffic profile (including the baseline profile) contains *WT*, *TT* or *TL* values of all the passengers. For example, for the WT QoS_{indivP} robustness of the dispatcher under us_2, if six of the resulting p-values of all the ten comparisons (corresponding to the $NR = 10$ generated uncertain traffic profiles) are less than 0.05, the WT QoS_{indivP} robustness $Robustness^{QoSi}_{1,us_2}$ is then quantified as six.

A.2 Evaluation metrics

Definition 8 (QoS_{allP} based Dispatcher Ranking). *To measure the difference of the 90 dispatchers, we first quantify their robustness under each uncertain situation in terms of each QoS_{allP} metric (with Eq. 5), and all the QoS_{allP} metrics with Eq. 7:*

$$OneDispatcherRobustness^{AllQoSa}_{us_o} = \sum_{l=1}^{6} Robustness^{QoSa}_{l,us_o} \tag{7}$$

We then perform the SD-G test to rank all the 90 dispatchers across all the uncertain situations in terms of each QoS_{allP} metric and all the QoS_{allP} metrics with Eq. 8 and Eq. 9, respectively:

$$\boldsymbol{DispatcherRank^{QoSa}_{l}} = SD - Gtest(\boldsymbol{D^{QoSa}_{1,l}}, \boldsymbol{D^{QoSa}_{2,l}}, \cdots, \boldsymbol{D^{QoSa}_{90,l}}) \tag{8}$$

$$\boldsymbol{DispatcherRank^{AllQoSa}} = SD - Gtest(\boldsymbol{D^{AllQoSa}_{1}}, \boldsymbol{D^{AllQoSa}_{2}}, \cdots, \boldsymbol{D^{AllQoSa}_{90}}) \tag{9}$$

where, $\boldsymbol{D^{QoSa}_{m,l}} = \left(Robustness^{QoSa}_{m,l,us_1} \cdots Robustness^{QoSa}_{m,l,us_{15}}\right)^{\top}$ *is the QoS_{allP} robustness of the mth dispatcher under each uncertain situation in terms of the lth QoS_{allP} metric;* $\boldsymbol{D^{AllQoSa}_{m}} = \left(OneDispatcherRobustness^{AllQoSa}_{m,us_1} \cdots OneDispatcherRobustness^{AllQoSa}_{m,us_{15}}\right)^{\top}$ *is the QoS_{allP} robustness of the mth dispatcher under each uncertain situation in terms of all the QoS_{allP} metrics.*

Definition 9 (QoS_{allP} based Uncertain Situation Ranking). *The QoS_{allP} robustness of all the dispatchers under each uncertain situation in terms of the lth QoS_{allP} metric and all the QoS_{allP} metrics combined are calculated with Eq. 10 and Eq. 11, respectively:*

$$AllDispatchersRobustness_{l,us_o}^{QoSa} = \sum_{m=1}^{90} Robustness_{m,l,us_o}^{QoSa} \tag{10}$$

$$AllDispatchersRobustness_{us_o}^{AllQoSa} = \sum_{l=1}^{6} AllDispatchersRobustness_{l,us_o}^{QoSa} \tag{11}$$

where $Robustness_{m,l,us_o}^{QoSa}$ is the quantified QoS_{allP} robustness of the mth dispatcher under the oth uncertain situation in terms of the lth QoS_{allP} metric (see Eq. 5). To rank the uncertain situations in terms of each QoS_{allP} metric across all the dispatchers, we perform the SD-G test with Eq. 12:

$$\boldsymbol{UncertainSituationRank_l^{QoSa}} = SD - Gtest(\boldsymbol{US_{1,l}^{QoSa}}, \boldsymbol{US_{2,l}^{QoSa}}, \cdots, \boldsymbol{US_{15,l}^{QoSa}}) \tag{12}$$

where $\boldsymbol{US_{o,l}^{QoSa}} = \left(Robustness_{1,l,us_o}^{QoSa} \cdots Robustness_{90,l,us_o}^{QoSa}\right)^\top$ consists of the QoS_{allP} robustness under the oth uncertain situation in terms of the lth QoS_{allP} metric of each dispatcher. The comprehensive ranking of the uncertain situations concerning all the QoS_{allP} metrics can be obtained with Eq. 13:

$$\boldsymbol{UncertainSituationRank^{AllQoSa}} = SD - Gtest(\boldsymbol{US_1^{AllQoSa}}, \boldsymbol{US_2^{AllQoSa}}, \cdots, \boldsymbol{US_{15}^{AllQoSa}}) \tag{13}$$

where $\boldsymbol{US_o^{AllQoSa}} = \left(OneDispatcherRobustness_{1,us_o}^{AllQoSa} \cdots OneDispatcherRobustness_{90,us_o}^{AllQoSa}\right)^\top$ composed of the QoS_{allP} robustness under the oth uncertain situation in terms of all the QoS_{allP} metrics of all the 90 dispatchers, each of which can be calculated with Eq. 7.

Definition 10 (QoS_{allP} Ranking). *Each dispatcher's robustness and all the dispatchers' robustness under all the 15 uncertain situations in terms of each QoS_{allP} metric are quantified with Eq.14 and Eq.15, respectively:*

$$OneDispatcherRobustness_l^{QoSa} = \sum_{o=1}^{15} Robustness_{l,us_o}^{QoSa} \tag{14}$$

$$AllDispatchersRobustness_l^{QoSa} = \sum_{m=1}^{90} OneDispatcherRobustness_{m,l}^{QoSa} \tag{15}$$

where $Robustness_{l,us_o}^{QoSa}$ is the quantified QoS_{allP} robustness of a dispatcher under the oth uncertain situation in terms of the lth QoS_{allP} metric (Eq. 5); the m represents the mth dispatcher. With this metric, we aim to obtain a high-level overview of the impact of the uncertain situations on each QoS_{allP} metric. We further rank the QoS_{allP} metrics with the SD-G test (Section 3.4), as defined in Eq. 16:

$$\boldsymbol{QoS_{allP}Rank} = SD - Gtest\ (\boldsymbol{QoSa_1}, \boldsymbol{QoSa_2}, \cdots, \boldsymbol{QoSa_6}) \tag{16}$$

where $\boldsymbol{QoSa_l} = \left(OneDispatcherRobustness_{1,l}^{QoSa} \cdots OneDispatcherRobustness_{90,l}^{QoSa}\right)^\top$ is the column vector composed of values of the robustness in terms of the lth QoS_{allP} metric of each of the 90 dispatchers under all the uncertain situations. Notice that, to compare with AllDispatchersRobustness (Eq. 15), this ranking is performed at the individual dispatcher level.

Definition 11 (QoS_{indivP} based Dispatcher Ranking). *The robustness of each dispatcher under each uncertain situation in terms of each QoS_{indivP} metric and all the three QoS_{indivP} metrics can be measured with Eq. 6 and Eq. 17, respectively.*

$$OneDispatcherRobustness_{us_o}^{AllQoSi} = \sum_{s=1}^{3} Robustness_{s,us_o}^{QoSi} \tag{17}$$

Based on the quantified QoS_{indivP} robustness of each dispatcher under each uncertain situation, we perform the SD-G test to rank all the dispatchers across all the 15 uncertain situations in terms of each QoS_{indivP} metric with Eq. 18, and all the QoS_{indivP} metrics with Eq. 19.

$$\boldsymbol{DispatcherRank_{s}^{QoSi}} = SD - Gtest(\boldsymbol{D_{1,s}^{QoSi}}, \boldsymbol{D_{2,s}^{QoSi}}, \cdots, \boldsymbol{D_{90,s}^{QoSi}}) \tag{18}$$

$$\boldsymbol{DispatcherRank^{AllQoSi}} = SD - Gtest(\boldsymbol{D_{1}^{AllQoSi}}, \boldsymbol{D_{2}^{AllQoSi}}, \cdots, \boldsymbol{D_{90}^{AllQoSi}}) \tag{19}$$

where, $\boldsymbol{D_{m,s}^{QoSi}}$ and $\boldsymbol{D_{m}^{AllQoSi}}$ are the quantified robustness of the mth dispatcher in terms of the sth QoS_{indivP} and all the QoS_{indivP} metrics under each uncertain situation, represented as $\boldsymbol{D_{m,s}^{QoSi}} = \left(Robustness_{m,s,us_1}^{QoSi} \cdots Robustness_{m,s,us_{15}}^{QoSi}\right)^{\top}$, and $\boldsymbol{D_{m}^{AllQoSi}} = \left(OneDispatcherRobustness_{m,us_1}^{AllQoSi} \cdots OneDispatcherRobustness_{m,us_{15}}^{AllQoSi}\right)^{\top}$, respectively.

Definition 12 (QoS_{indivP} based Uncertain Situation Ranking). *The QoS_{indivP} robustness of all the dispatchers under each uncertain situation in terms of each QoS_{indivP} metric and all the three QoS_{indivP} metrics combined can be measured with Eq. 20 and Eq. 21, respectively.*

$$AllDispatchersRobustness_{s,us_o}^{QoSi} = \sum_{m=1}^{90} Robustness_{m,s,us_o}^{QoSi} \tag{20}$$

$$AllDispatchersRobustness_{us_o}^{AllQoSi} = \sum_{s=1}^{3} AllDispatchersRobustness_{s,us_o}^{QoSi} \tag{21}$$

where $Robustness_{m,s,us_o}^{QoSi}$ is the QoS_{indivP} robustness (Eq. 6) of the mth dispatcher under the oth uncertain situation in terms of the sth QoS_{indivP} metric. To study the QoS_{indivP} specific ranking of the uncertain situations across all the dispatchers, we perform the SD-G test with Eq. 22 shown below:

$$\boldsymbol{USRank_{s}^{QoSi}} = SD - Gtest(\boldsymbol{US_{1,s}^{QoSi}}, \boldsymbol{US_{2,s}^{QoSi}}, \cdots, \boldsymbol{US_{15,s}^{QoSi}}) \tag{22}$$

where $\boldsymbol{US_{o,s}^{QoSi}} = \left(Robustness_{1,s,us_o}^{QoSi} \cdots Robustness_{90,s,us_o}^{QoSi}\right)^{\top}$ contains each dispatcher's robustness under the oth uncertain situation in terms of the sth QoS_{indivP} metric. The comprehensive ranking of considering all the QoS_{indivP} metrics can be performed with Eq. 23:

$$\boldsymbol{USRank^{AllQoSi}} = SD - Gtest(\boldsymbol{US_{1}^{AllQoSi}}, \boldsymbol{US_{2}^{AllQoSi}}, \cdots, \boldsymbol{US_{15}^{AllQoSi}}) \tag{23}$$

where $\boldsymbol{US_{o}^{AllQoSi}} = \left(OneDispatcherRobustness_{1,us_o}^{AllQoSi} \cdots OneDispatcherRobustness_{90,us_o}^{AllQoSi}\right)^{\top}$, $OneDispatcherRobustness_{m,us_o}^{AllQoSi}$ is the quantified robustness of the mth dispatcher under the oth uncertain situation in terms of all the QoS_{indivP} metrics, which can be obtained with Eq. 17.

Definition 13 (QoS_{indivP} Ranking). *The QoS_{indivP} robustness of each and all the dispatchers under all the uncertain situations can be measured with Eq 24 and Eq 25, respectively:*

$$OneDispatcherRobustness_{s}^{QoSi} = \sum_{o=1}^{15} Robustness_{s,us_o}^{QoSi} \tag{24}$$

$$AllDispatchersRobustness_{s}^{QoSi} = \sum_{m=1}^{90} OneDispatcherRobustness_{m,s}^{QoSi} \tag{25}$$

where $Robustness_{s,us_o}^{QoSi}$ is the sth QoS_{indivP} robustness of a dispatcher under the oth uncertain situation (see Eq. 6 for its calculation). We further rank the QoS_{indivP} metrics across all the dispatchers under all *the uncertain situations with the SD-G test, as shown in Eq. 26 below:*

$$\boldsymbol{QoS_{indivP}Rank} = SD - Gtest(\boldsymbol{QoSi_1}, \boldsymbol{QoSi_2}, \boldsymbol{QoSi_3}) \tag{26}$$

where $\boldsymbol{QoSi_s} = \left(OneDispatcherRobustness_{1,s}^{QoSi} \cdots OneDispatcherRobustness_{90,s}^{QoSi}\right)^{\top}$ *is the column vector composed of the robustness values of each of the 90 dispatchers in terms of the* s*th* QoS_{indivP} *metric.*

Definition 14 (Total robustness). *We use Eq. 28 to measure the robustness of all the dispatchers under all the 15 uncertain situations in terms of all the six* QoS_{allP} *metrics :*

$$OneDispatcherRobustness^{AllQoSa\&AllUS} = \sum_{l=1}^{6}\sum_{o=1}^{15} Robustness_{l,us_o}^{QoSa} \tag{27}$$

$$AllDispatchersRobustness^{AllQoSa\&AllUS} = \sum_{m=1}^{90} OneDispatcherRobustness_{m}^{AllQoSa\&AllUS} \tag{28}$$

Similarly, the QoS_{indivP} *robustness of all the dispatchers under all the 15 uncertain situations in terms of all the three* QoS_{indivP} *metrics are quantified with Eq. 30:*

$$OneDispatcherRobustness^{AllQoSi\&AllUS} = \sum_{s=1}^{3}\sum_{o=1}^{15} Robustness_{s,us_o}^{QoSi} \tag{29}$$

$$AllDispatchersRobustness^{AllQoSi\&AllUS} = \sum_{m=1}^{90} OneDispatcherRobustness_{m}^{AllQoSi\&AllUS} \tag{30}$$

B APPENDIX – DETAILED RESULTS

Table 12. QoS_{allP} robustness of the 90 dispatchers under each uncertain situation (*UncerSitua*) - RQ1.2. For each uncertain situation, there are, in total, 90 (dispatchers) p-values for each QoS_{allP} metric (e.g., columns *AWT* and *LWT*) and 90 (dispatchers) × 6 (QoS_{allP}) = 540 p-values for all the QoS_{allP} metrics together (column *Overall*). Each *Count* cell of each QoS_{allP} column and the *Overall* column tells the count calculated with Eq. 10 or Eq. 11, respectively. A lower count indicates a higher robustness under a specific uncertain situation.

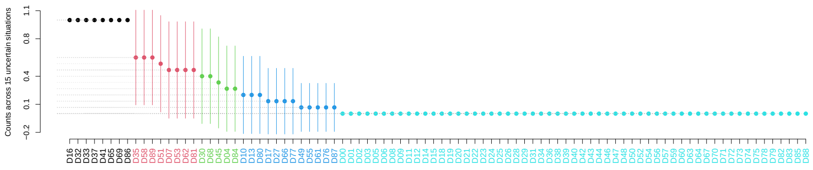
AWT		LWT		ATT		LTT		ATD		LTD		Overall	
UncerSitua	Count	UncerSitua	Count	UncerSitua	Count	UncerSitu	Count	UncerSitua	Count	UncerSitua	Count	UncerSitua	Count
LunchPeak													
usL	20	usM-C-L	30	usM	39	usM	44	usM-C	31	usC	37	usC	183
usC	19	usC	29	usM-C	38	usC-L	38	usC	30	usM-C	35	usM-C	183
usM-U	18	usM-C-U	29	usC	35	usM-C	35	usM-U	28	usM-C-L	34	usM	178
usC-U	17	usC-U	27	usC-U	33	usC	33	usC-U	25	usM-C-U	34	usM-U	165
usM-C	17	usM-C	27	usM-U	33	usM-U	33	usM	23	usM-L	32	usM-C-L	159
usM	16	usM	26	usC-L	30	usM-C-L	33	usM-C-L	21	usC-U	31	usC-U	152
usL-U	15	usM-U	26	usU	29	usM-C-L-U	32	usM-C-U	18	usM	30	usC-L	148
usM-C-U	15	usL	25	usM-L	28	usM-L	31	usU	16	usC-L	28	usM-L	142
usU	14	usM-C-L-U	25	usM-C-L	28	usM-L-U	30	usM-L	15	usC-L-U	28	usM-C-U	142
usC-L	14	usC-L	24	usM-C-U	27	usC-L-U	29	usC-L	14	usM-U	27	usM-C-L-U	128
usM-C-L-U	14	usL-U	24	usL	26	usU	23	usL-U	12	usM-C-L-U	26	usC-L-U	127
usC-L-U	13	usM-L	24	usC-L-U	25	usL	20	usM-L-U	11	usU	22	usU	126
usM-C-L	13	usC-L-U	24	usM-C-L-U	23	usC-U	19	usL	10	usM-L-U	22	usL	121
usM-L-U	13	usU	22	usL-U	20	usM-C-U	19	usC-L-U	8	usL	20	usM-L-U	115
usM-L	12	usM-L-U	20	usM-L-U	19	usL-U	12	usM-C-L-U	8	usL-U	20	usL-U	103
UpPeak													
usL	36	usC-U	52	usM-C	35	usC-L	77	usM	35	usC-L	44	usC	236
usC	32	usC-L	51	usC	32	usM-C	67	usC	29	usC	42	usM-C	217
usL-U	32	usM-C	49	usM	28	usC	55	usL	17	usM-C-U	39	usC-L	211
usM-C-U	32	usC	46	usC-U	25	usM-C-L	55	usM-C-L	17	usM-C-L	38	usM-C-L	209
usM-C-L	30	usM-C-L	46	usM-C-L	23	usM-C-L-U	53	usU	16	usC-U	37	usM-C-U	201
usM	28	usM-C-U	46	usC-L	20	usC-U	52	usC-U	14	usM-C	37	usM	197
usU	19	usM-C-L-U	44	usM-C-U	20	usM-C-U	51	usL-U	14	usM	34	usC-U	191
usM-C	15	usL	39	usM-L	17	usC-L-U	47	usM-C	14	usL	33	usM-C-L-U	164
usM-L-U	13	usM	28	usM-U	15	usM	44	usM-L	14	usM-L	33	usL	161
usM-U	12	usL-U	23	usC-L-U	14	usM-L	42	usM-U	13	usM-C-L-U	33	usM-L	126
usC-U	11	usC-L-U	19	usM-C-L-U	14	usM-U	37	usM-C-U	13	usL-U	32	usC-L-U	118
usM-L	11	usM-L-U	13	usL	8	usL	28	usC-L	10	usM-L-U	32	usL-U	116
usM-C-L-U	11	usU	9	usU	8	usM-L-U	27	usC-L-U	10	usU	27	usM-U	111
usC-L-U	10	usM-L	9	usL-U	6	usU	20	usM-L-U	9	usM-U	26	usM-L-U	100
usC-L	9	usM-U	8	usM-L-U	6	usL-U	9	usM-C-L-U	9	usC-L-U	18	usU	99

M: Mass C: Capacity Factor L: Loading Time U: Unloading Time
us∗: uncertain situation caused by uncertain factor ∗. For example, usM-L represents the uncertain situation caused by Mass and Loading Time.

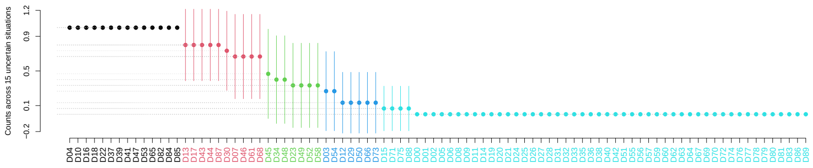
Table 13. QoS_{indivP} robustness of the 90 dispatchers under each uncertain situation (*UncerSitua*) - RQ2.2. For each uncertain situation, there are, in total, 90 (dispatchers) × 10 (uncertain traffic profiles) = 900 p-values for each QoS_{indivP} metric (e.g., *WT* column) and 90 (dispatchers) × 10 (uncertain traffic profiles) × 3 (QoS_{indivP}) = 2700 p-values for all the QoS_{indivP} metrics together (*Overall* column). Each *Count* cell of each QoS_{indivP} metric column and the *Overall* column tells the count calculated with Eq. 20 or Eq. 21, respectively. A lower count indicates a higher robustness under a specific uncertain situation.

WT		TT		TD		Overall	
UncerSitua	Count	UncerSitua	Count	UncerSitua	Count	UncerSitua	Count
LunchPeak							
usL	157	usM-C	278	usM-C	197	usM-C	615
usC	156	usM	255	usC	185	usC	594
usM-C-U	150	usC	253	usM	173	usM	567
usM-U	147	usM-U	226	usM-U	147	usM-U	520
usL-U	146	usU	217	usC-U	144	usU	491
usM-C	140	usC-U	198	usU	137	usC-U	475
usM	139	usC-L	184	usM-C-U	127	usM-C-U	448
usU	137	usM-C-L	183	usC-L	120	usC-L	429
usC-U	133	usM-C-U	171	usM-L	120	usM-C-L	404
usM-L	127	usM-L	156	usL	109	usM-L	403
usC-L	125	usM-L-U	148	usM-C-L	101	usL	391
usM-C-L	120	usL	125	usM-L-U	100	usM-L-U	358
usC-L-U	110	usM-C-L-U	120	usL-U	88	usL-U	339
usM-L-U	110	usC-L-U	109	usM-C-L-U	67	usM-C-L-U	289
usM-C-L-U	102	usL-U	105	usC-L-U	65	usC-L-U	284
UpPeak							
usL	438	usM-C	492	usM	384	usM	1137
usL-U	390	usC-U	415	usC	343	usC	1065
usU	386	usC	396	usC-U	333	usC-U	1027
usM	361	usM	392	usM-C	294	usM-C	984
usM-C-U	331	usM-C-U	304	usM-C-U	249	usM-C-U	884
usC	326	usM-C-L	284	usL	240	usM-C-L	809
usM-L-U	318	usC-L	252	usM-C-L	226	usL	702
usM-C-L	299	usM-U	197	usU	202	usM-L	655
usC-U	279	usM-L	189	usM-L	196	usC-L	638
usM-L	270	usM-C-L-U	175	usC-L	192	usU	636
usC-L-U	223	usC-L-U	156	usM-L-U	185	usM-L-U	596
usM-C-L-U	209	usM-L-U	93	usM-U	177	usM-U	581
usM-U	207	usU	48	usL-U	169	usL-U	566
usM-C	198	usL	24	usC-L-U	145	usM-C-L-U	528
usC-L	194	usL-U	7	usM-C-L-U	144	usC-L-U	524

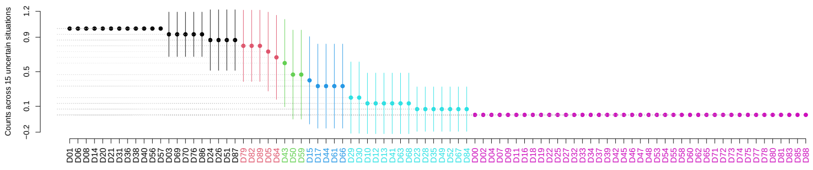
M: Mass C: Capacity Factor L: Loading Time U: Unloading Time
us∗: uncertain situation caused by uncertain factor ∗. For example, usM-L represents the uncertain situation caused by Mass and Loading Time.



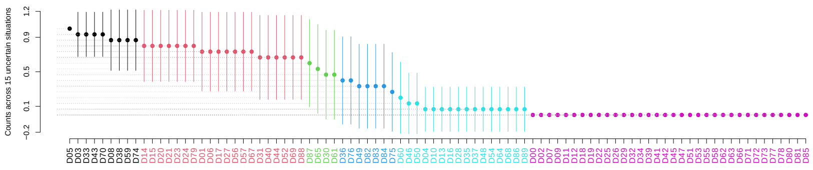
(a) *LunchPeak-AWT*.



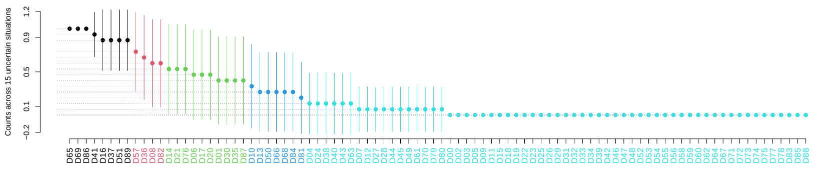
(b) *LunchPeak-LWT*.



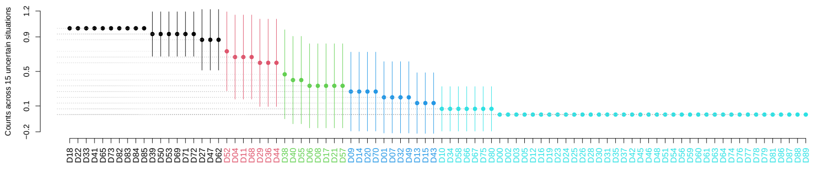
(c) *LunchPeak-ATT*.



(d) *LunchPeak-LTT*.

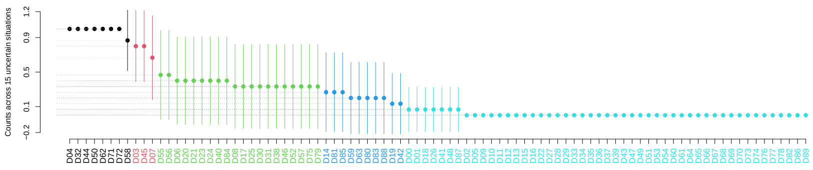


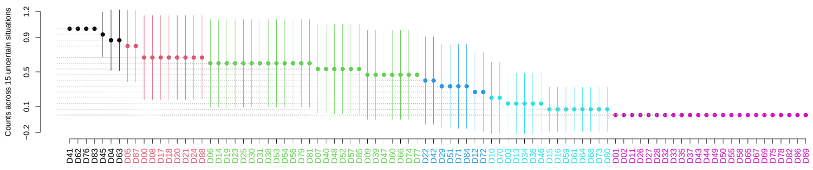
(e) *LunchPeak-ATD*.

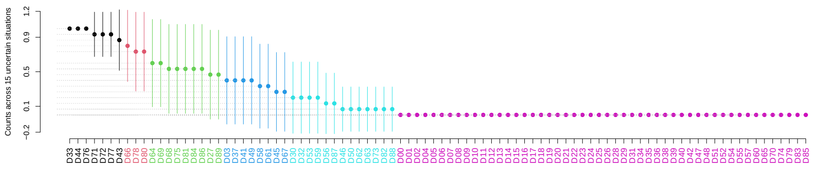


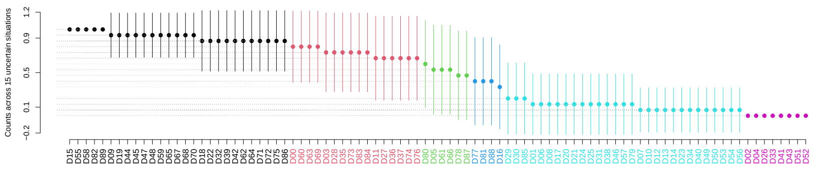
(f) *LunchPeak-LTD*.

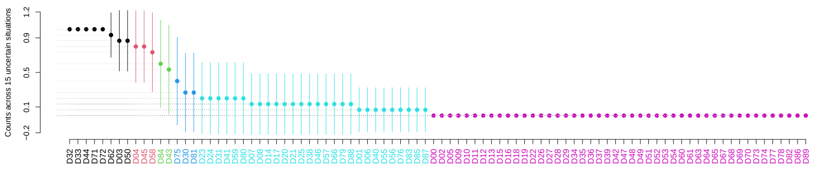
Fig. 14. The *SD-G* ranking of the 90 dispatchers across the 15 uncertain situations in terms of each QoS_{allP} metric - RQ1.1. The x-axis shows the 90 dispatchers sorted according to their counts under all the uncertain situations in terms of a specific QoS_{allP} metric, whereas the same color of the adjacent dispatchers indicates the same group, implying that the difference of the QoS_{allP} specific robustness of these dispatchers across the 15 uncertain situations are negligible. Each dot represents the average count across the 15 uncertain situations on each sample. The length of each line indicates twice of the standard deviation of each sample.


(g) *UpPeak-AWT*.


(h) *UpPeak-LWT*.


(i) *UpPeak-ATT*.


(j) *UpPeak-LTT*.


(k) *UpPeak-ATD*.

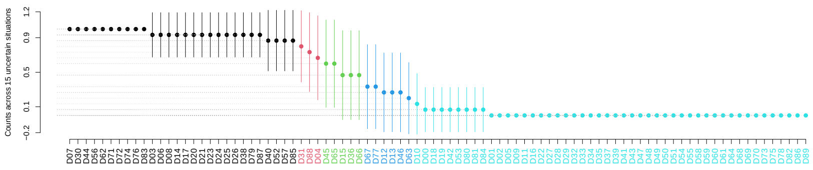

(l) *UpPeak-LTD*.

Fig. 14. The *SD-G* ranking of the 90 dispatchers across the 15 uncertain situations in terms of each QoS_{allP} metric - RQ1.1 (continued)

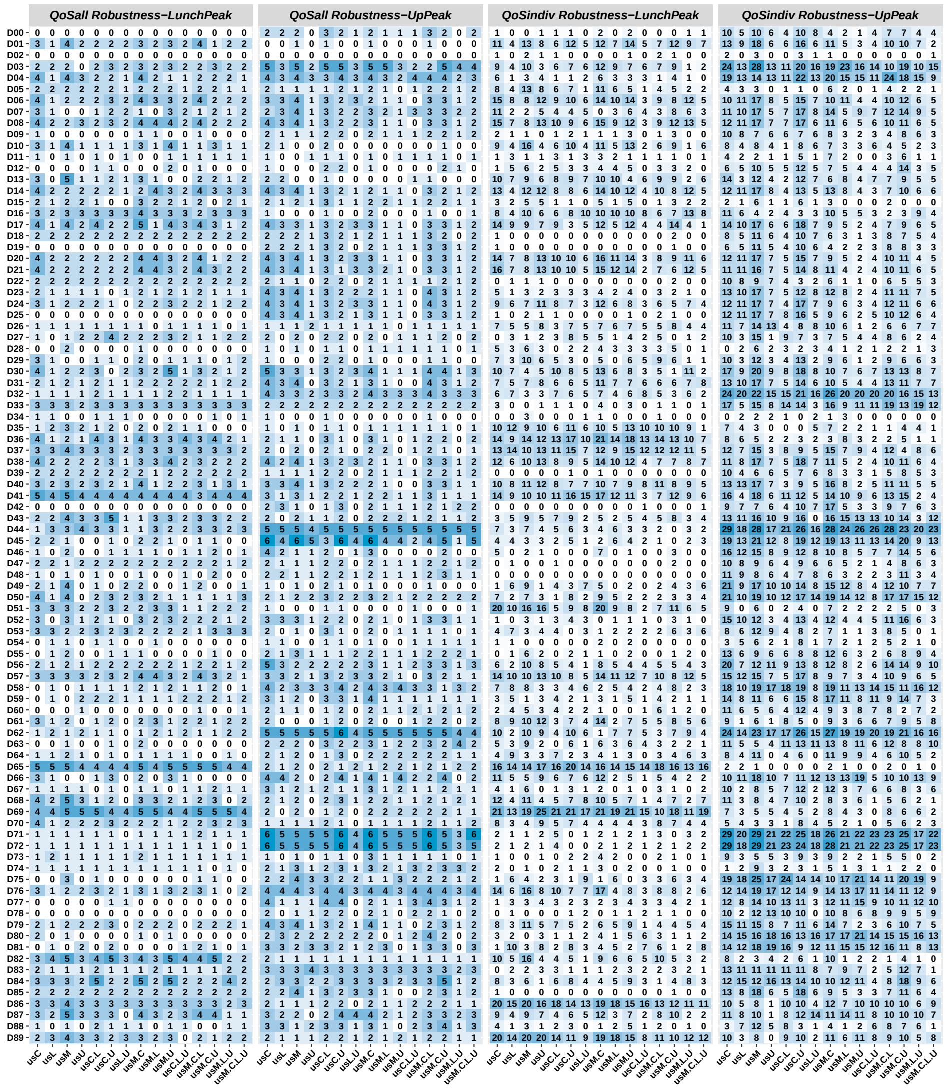
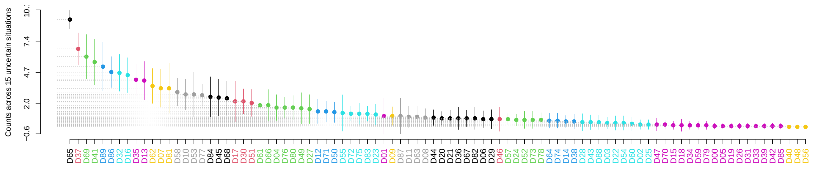
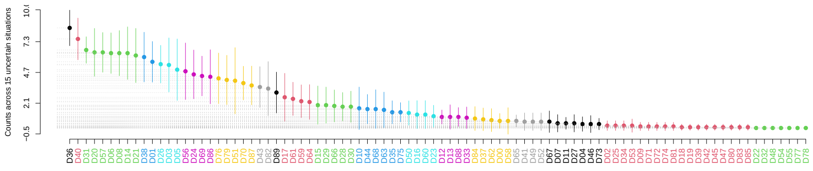


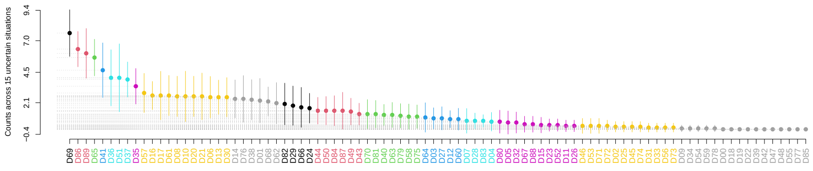
Fig. 15. QoS_{allP} and QoS_{indivP} robustness under each uncertain situation - RQ1.1 & RQ2.1. Each row in each subplot show the QoS_{allP} (the left two subplots) or QoS_{indivP} (the right two subplots) robustness of one dispatcher under each uncertain situation. Each grid is a visualization of the quantified robustness of a dispatcher under a specific uncertain situation, computed with Eq. 7 (for QoS_{allP}) or Eq. 17 (for QoS_{indivP}). The bluer the grid, the higher the count and the less robust a dispatcher performed. For QoS_{allP} robustness, the value in a grid ranges from 0 to 6 (corresponding to the 6 QoS_{allP} metrics). For QoS_{indivP} robustness, the value in a grid ranges from 0 to 30 (3 QoS_{indivP} metrics, each of which has 10 p-values).



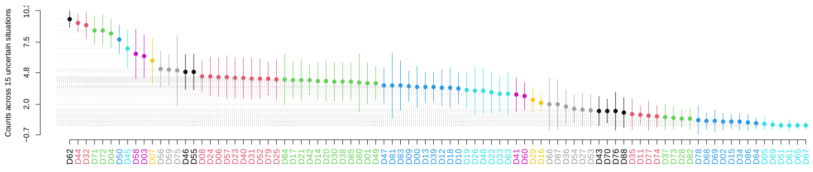
(a) *LunchPeak-WT*.



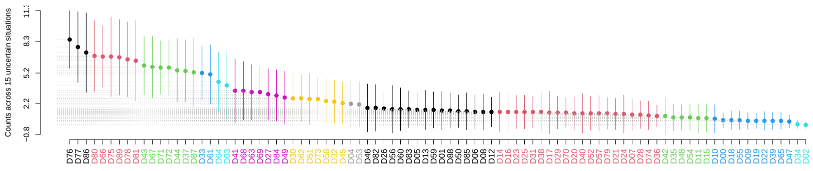
(b) *LunchPeak-TT*.



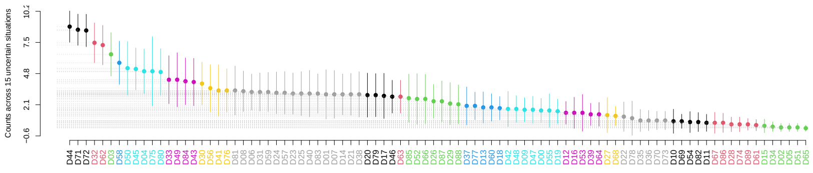
(c) *LunchPeak-TD*.



(d) *UpPeak-WT*.



(e) *UpPeak-TT*.



(f) *UpPeak-TD*.

Fig. 16. The *SD-G* ranking of the 90 dispatchers across the 15 uncertain situations in terms of each QoS_{indivP} metric - RQ2.1. The x-axis shows the 90 dispatchers sorted according to their counts under all the uncertain situations in terms of a specific QoS_{indivP} metric, whereas the same color of the adjacent dispatchers indicates the same group, implying that the difference of the QoS_{indivP} specific robustness of these dispatchers across the 15 uncertain situations are negligible. Each dot represents the average count across the 15 uncertain situations on each sample, and the length of each line indicates twice of the standard deviation of each sample.