How to Measure the Robustness of Shunting Plans

Roel van den Broek
Department of Computer Science, Utrecht University
Utrecht, The Netherlands
r.w.vandenbroek@uu.nl

Han Hoogeveen
Department of Computer Science, Utrecht University
Utrecht, The Netherlands
j.a.hoogeveen@uu.nl

Marjan van den Akker
Department of Computer Science, Utrecht University
Utrecht, The Netherlands
j.m.vandenakker@uu.nl

Abstract

The general problem of scheduling activities subject to temporal and resource constraints as well as a deadline emerges naturally in numerous application domains such as project management, production planning, and public transport. The schedules often have to be implemented in an uncertain environment, where disturbances cause deviations in the duration, release date or deadline of activities. Since these disruptions are not known in the planning phase, we must have schedules that are robust, i.e., capable of absorbing the disturbances without large deteriorations of the solution quality. Due to the complexity of computing the robustness of a schedule directly, many surrogate robustness measures have been proposed in literature. In this paper, we propose new robustness measures, and compare these and several existing measures with the results of a simulation study to determine which measures can be applied in practice to obtain good approximations of the true robustness of a schedule with deadlines. The experiments are performed on schedules generated for real-world scheduling problems at the shunting yards of the Dutch Railways (NS).

2012 ACM Subject Classification Computing methodologies → Planning under uncertainty

Keywords and phrases robustness, resource-constrained project scheduling, partial order schedule, uncertainty, Monte Carlo simulation, train shunting

Digital Object Identifier 10.4230/OASIcs.ATMOS.2018.3

1 Introduction

For the shunting yards operated by the *Dutch Railways (NS)*, the largest passenger railway operator in the Netherlands, human planners create daily *shunting plans* describing all the activities that have to be performed, such as cleaning, maintenance, parking, and movements of trains. The objective for the planners is to construct a schedule in which all service activities on a train are completed before its deadline, which is the scheduled departure time. However, since arriving trains might be delayed, and performing a service activity can take longer than expected in practice, a shunting plan that is feasible with respect to the nominal timetable and activity durations may become infeasible during operation if a departure from the shunting yard is delayed due to disturbances.

18th Workshop on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems (ATMOS 2018).
Editors: Ralf Borndörfer and Sabine Storandt; Article No. 3; pp. 3:1–3:13

OpenAccess Series in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Since the exact disturbances that will occur during the execution of a schedule are not known in advance, a practical strategy to handle the uncertainty is to construct a baseline schedule and a simple scheduling policy that adapts the initial schedule to the disruptions. The baseline schedule of a shunting plan of the NS is a partial order schedule of all activities, with precedence relations to ensure that any execution of the plan is *resource feasible*. The scheduling policy for the shunting plans is the *Earliest Start Time (EST)* or *Right Shift* policy, which assigns activities to their earliest start time in the baseline schedule, and, in case of disruptions during operation, delays activities that have not yet started as much as necessary while maintaining the ordering in the baseline schedule. Operational disturbances that cannot be absorbed by the shunting plan with the EST policy have to be handled by the human planners. The Dutch Railways prefers robust shunting plans that require little rescheduling during the operational phase, as the ad-hoc modifications to the schedule made by human planners often have a cascading effect in other parts of the shunting plan. A quantitative metric of this preference is the probability that the execution of a baseline plan will result in a delayed train departure.

In order to find robust baseline schedules for scheduling problems with deadlines, we have to determine a priori whether a schedule will perform well in the uncertain operational environment. However, the robustness of a schedule depends heavily on the available knowledge of the uncertainty: which elements in the scheduling problem can be disrupted by uncontrollable factors, and what are the distributions of those events? Often, the uncertain elements are known during the planning stage, but data on the distribution of the uncertainty are lacking. As a result, the robustness of a schedule is hard to compute in general, and assumptions on the uncertainty have to be made.

An approach often used to estimate the robustness is to simulate the performance of the schedule in many different scenarios sampled from the (assumed) distributions of the uncertainty. Although simulation is a powerful and versatile tool that gives an accurate estimate if a sufficient number of samples is used, it tends to be a computationally expensive technique. As solution methods for scheduling problems typically evaluate a large number of schedules to find the (near-)optimal solution, using simulation as a subroutine in the solution method might not be feasible. Therefore, several robustness measures that act as a surrogate for the sampled robustness of a schedule have been developed in the past few decades.

The contribution of this paper is to identify robustness measures that both properly predict the robustness of a schedule subject to deadlines, and can be evaluated efficiently. To this end, we generate a large number of schedules for real-world instances of the shunting yards operated by the NS. We perform a Monte Carlo simulation of schedules with uncertainty to obtain a good approximation of the robustness, and compare the outcome with the predictions of the robustness measures to determine if any of the estimations show a strong correlation with the sampled robustness. We base the comparison on two performance metrics, which are the fraction of delayed schedules, and the average lateness of the schedules.

The remainder of this paper is organized as follows. We start in Section 2 with a review of related work on robustness in resource-constrained project management, followed by a brief summary of the common concepts and notation in this paper in Section 3. We provide in Section 4 an overview of several robustness measures from literature, and propose some new, path-based measures. The instances provided by the NS as well as the setup of the Monte Carlo simulation study are discussed in Section 5. We compare the predictions of the robustness measures with the results of the simulation study in Section 6, and finish with concluding remarks and potential directions for further research in Section 7.

2 Literature overview

Most of the robustness measures proposed in literature are for resource-constrained project scheduling problems, where the standard objective is to minimize the makespan of the schedule. These measures are mainly based on the concept of *slack*. The *total slack* is defined as the the maximum amount of time that we can delay an activity without increasing the makespan of the total schedule, whereas *free slack* is the amount of time by which an activity can be delayed without delaying any other activity in the schedule.

A simple slack-based robustness estimation, proposed by [6], is to compute the average of the total slacks of all activities. By simulating many realizations of job shop schedules, [6] showed that a large percentage of the variation in the realized makespan was explained by the average slack of the schedule. Similarly, [1] proposed the sum of free slacks as a robustness measure.

Based on the observation that, in addition to the total amount of slack, the distribution of the slack over the schedule affects the robustness as well, [3] proposed several variants of the sum of free slacks. These robustness measures weigh the free slack by the number of successors, and substitute the free slack with a binary slack indicator function or an upper bound on the slack based on the activity duration.

The relation of a number of existing and newly proposed robustness measures to the fraction of feasible schedule realizations in a Monte Carlo simulation has been investigated by [5]. For instances of the discrete time/cost trade-off problem, they reported high values (> 0.91) of the coefficient of determination for the sum of total slacks measure and successor-weighted variants of it.

A similar comparison of robustness metrics in a Monte Carlo simulation was performed by [2]. In contrast to the work of [5], their results showed that summing the unweighted slack of the activities has at best a weak correlation with the expected makespan of the schedule.

When scheduling activities subject to deadlines, the primary objective is to find a feasible schedule. However, the concepts of free and total slack do not fully capture the slack of a schedule with respect to its deadline. To quantify this type of slack, we can view a schedule with deadlines as a special case of a *Simple Temporal Network (STN)*, which is a directed graph with both minimum and maximum time lags on the arcs that was introduced by [4]. For this type of graph, there are *flexibility metrics* that aggregate the slack with respect to all the temporal constraints, including the deadlines. The *naive flexibility* of an STN is the sum of the difference between the latest and earliest start time of each activity, i.e., the total slack relative to the deadline instead of the makespan of the schedule. Analog to the free slack of an activity, [15] proposed the *concurrent flexibility* metric, which is based on interval schedules. An interval schedule specifies for each activity an interval such that every activity can start at any time within its interval independently of the other events, and without exceeding the deadline of the schedule. The concurrent flexibility of an STN is defined as the maximal sum of the interval lengths over all possible interval schedules. A linear programming formulation was proposed by the authors to compute the concurrent flexibility. It was shown in [14] that a schedule with a high flexibility is not always robust to disruptions.

The limitations of the sum of free slacks metric were discussed by [8], and they proposed to use the minimum free slack over all activities as a robustness measure for schedules with a deadline, and provided an algorithm that maximizes the minimum free slack by distributing the free slack evenly over the schedule. Their approach is essentially the concurrent flexibility metric, proposed by [15], with as objective to maximize the minimum interval length instead of the sum of the intervals.

An extensive comparison of robustness measures can be found in the paper of [7]. They investigated the correlation between the surrogate robustness measures and the probability that the completion time of a schedule exceeds its nominal makespan, which was approximated using a Monte Carlo simulation. Their results showed that the strongest correlation ($R^2 > 0.64$) with the robustness performance metric in the simulation was achieved with a robustness measure that computes the *slack sufficiency*, which is based on the ratio between the free slack and the processing time of an activity.

Despite the many surrogate robustness measures in literature, there is no consensus on which of these provides a good approximation of the true robustness of a schedule. In the simulation studies of [6], [5], [2] and [7], only schedules without deadlines are considered, focusing mainly on the expected makespan and related performance metrics. However, a good expected makespan of a schedule constrained by a deadline does not necessarily imply that the schedule will respect its deadline. Therefore, additional research is required to verify their results for schedules with deadlines.

3 Preliminaries

In this paper, we consider the general resource-constrained scheduling problem with deadlines. For a scheduling problem with activities 1 to n, and a deadline at time T for all activities, we define a *baseline schedule* σ as a pair $(S_\sigma, \mathcal{POS}_\sigma)$, where $S_\sigma = \{0, \ldots, n+1\}$ is the *activity set*, and $\mathcal{POS}_\sigma$ a *partial order schedule* of these activities:

$$\mathcal{POS}_\sigma = \{i \prec j \mid \forall i, j \in S_\sigma : i \text{ directly precedes } j \text{ in } \sigma\}.$$

In this schedule, the activities 0 and $n+1$ are dummy activities representing the start and end of the schedule, respectively; the precedence relations needed to ensure that all activities take place between the start and the end activity are contained in the partial ordering.

Each activity i has a nominal processing time $p_i \in \mathbb{R}_+$, with $p_0 = p_{n+1} = 0$. We assume that the release date of each activity is equal to 0, and that all activities, in particular $n+1$, have to be finished before the deadline T. Note that scheduling problems with an individual release date or deadline of activity i can still be modeled in the schedule by adding a dummy activity between i and the start or end activity, respectively.

From the baseline schedule σ, we can compute for each activity i the time window in which it has to be processed. The *earliest start time* est_i^σ is the earliest possible time at which all predecessor activities can be finished. Similarly, the *latest finish time* lft_i^σ is equal to the latest possible completion time of activity i such that the schedule remains feasible with respect to the deadline. The *latest start time* lst_i^σ and *earliest finish time* eft_i^σ can be derived from the latest or earliest counterpart by subtracting or adding the processing time p_i, respectively. The earliest finish time of activity $n+1$ is known as the *makespan* or $C_{\max}$.

The concept of *slack* is commonly used to quantify the robustness of a schedule. We define the *total slack* ts_i^σ of activity i in schedule σ as the maximum amount of time by which we can delay the activity such that no deadlines are exceeded in the schedule. Equivalently, the slack is the difference between the earliest and latest start time of the activity, $ts_i^\sigma = lst_i^\sigma - est_i^\sigma$. Note that this definition differs slightly from the formulation given in 2, where the total slack is computed with respect to the makespan instead of the schedule deadline. However, for each activity in the schedule, the difference in slack between the two definitions is a constant, namely $T - C_{\max}$. A different type of slack is the *free slack* fs_i^σ, which is the maximum amount of time that activity i can be delayed without affecting any other activity. That is,

we define the free slack as

$$fs_i^\sigma = \min_{j \in succ_\sigma(i)} \{est_j\} - eft_i, \tag{1}$$

where $succ_\sigma(i)$ are the successor activities of i in the schedule.

An intuitive graph-based representation of the partial ordering can be constructed by modeling each activity i in activity set as a vertex v_i, and adding for every precedence relation $i \prec j \in \mathcal{POS}$ an arc from v_i to v_j to the graph. The result is a digraph $G_\sigma = (V_\sigma, A_\sigma)$, in which a directed path represents a set of activities that have to be performed sequentially. Similar to slack of an activity, we define the slack of a path $\pi = (\pi_1, \ldots, \pi_k)$ as

$$s_\pi^\sigma = lft_{\pi_k}^\sigma - est_{\pi_1}^\sigma - \sum_{i \in \pi} p_i,$$

which is the maximal amount of time we can delay activities on the path without exceeding the deadline of π_k.

Shunting plans

The schedules that we use our experiments are solutions to a scheduling problem of the Dutch Railways that arises at shunting yards, which are networks of tracks connected by switches that contain facilities providing services such as cleaning and maintenance to the trains. In this scheduling problem, which is a variant of the *Train Unit Shunting Problem* described in [13], we have a number of train units that arrive during the evening on the shunting yard. These arrivals happen according to a static timetable, which lists the arrival time as well as the train – a sequence of coupled train units – in which each train unit arrives. The train units have to leave the shunting yard the next morning, again based on the timetable. Note that the arrival train of a train unit is not necessarily the same as the departure train. During their stay at the shunting yard, the train units have to move through different facilities to receive service tasks such as cleaning and maintenance, and must be parked on an appropriate track to wait until departure.

The scheduling problem at the shunting yards is then to construct a *shunting plan*, which is a schedule that describes all the activities on the shunting yard such as coupling and decoupling train units, service tasks and train movements, such that the service tasks of each train unit in a departing train are completed before the departure time, and none of the resource capacity constraints are exceeded in the shunting plan. A more in-depth description of the scheduling problem can be found in [13].

4 Robustness measures

In this section we discuss the robustness measures that we will compare in our experiments. Surrogate robustness measures that rely heavily on the exact distribution of the uncertainty in a schedule might produce accurate predictions of the robustness, but their applicability to real-world scheduling problems is limited, since quantitative data of the uncertainty are often scarce in practice. Therefore, robustness measures with a low dependency of the available knowledge of the uncertainty are preferred.

Robustness measures are usually created with the assumption that the nominal or expected processing time of the activities is known. If a robustness measure does not rely on any other information about the uncertainty, the robustness is solely estimated from the structure of

the baseline schedule. The two robustness measures applied most often in literature, namely the sum of total slacks ([6]),

$$RM_1(\sigma) = \sum_i ts_i^\sigma, \tag{2}$$

and the sum of free slacks ([1]),

$$RM_2(\sigma) = \sum_i fs_i^\sigma, \tag{3}$$

are examples of measures that depend only on the nominal activity durations. Furthermore, the standard resource-constrained scheduling objective, the makespan or, equivalently, the minimum total slack,

$$RM_3(\sigma) = \min_i ts_i^\sigma, \tag{4}$$

can be viewed as an expectation-only robustness measure as well.

Another robustness measure of this type that was shown by [7] to provide good estimations of the robustness of a schedule was based on *slack sufficiency*, which compares the free slack of an activity to a fraction of the duration of that activity or one of its predecessors in the schedule. In the work of [7], this robustness measure is defined as

$$RM_4(\sigma) = \sum_i |\{j \mid j \in prec_\sigma(i) \cup \{i\}, fs_i \geq \lambda p_j\}| \tag{5}$$

where $prec_\sigma(i)$ are the predecessors of activity i in the schedule and $0 < \lambda < 1$. The authors suggested that λ should be set to the expected deviation from the nominal processing time of the activities due to disruptions.

A more complex robustness measure depending only on the expected activity duration is the interval schedule based approach of [15]. It finds the maximal assignment of intervals to activities such that each activity i can be scheduled within its interval (e_i, l_i) independently of the other activities. To achieve this, the intervals are computed with the linear program

$$\begin{aligned} RM_5(\sigma) = &\max \sum_i (l_i - e_i) \\ &\text{subject to} \\ &est_i^\sigma \leq e_i \leq s_i \leq lst_i^\sigma \quad \forall i \\ &l_i + p_i \leq e_j \qquad\qquad \forall i \prec j \in \mathcal{POS}_\sigma. \end{aligned} \tag{6}$$

As an alternative to solving this linear program, [10] formulated a matching problem based on the dual problem.

Analog to the work of [8], we can change the objective of the linear program of [15] to maximize the minimum interval, which will result in a more evenly distributed interval schedule. The linear program then becomes

$$\begin{aligned} RM_6(\sigma) = &\max \min_i (l_i - e_i) \\ &\text{subject to} \\ &est_i^\sigma \leq e_i \leq l_i \leq lst_i^\sigma \quad \forall i \\ &l_i + p_i \leq e_j \qquad\qquad \forall i \prec j \in \mathcal{POS}_\sigma. \end{aligned} \tag{7}$$

In contrast to the previous surrogate robustness measures, the measure of [15] focuses on the entire graph structure instead of just the slack of the individual activities. However, an optimal solution to the linear program is an interval schedule that assigns large intervals to concurrent activities, and, consequently, only small intervals to sequential activities, thus overemphasizing parallel activities.

A compromise between activity-based and schedule-based measures is to predict the robustness of a schedule from the paths in the partial ordering. Without any knowledge of the uncertainty, it is reasonable to assume that the likelihood of a disruption on a path increases with the number of activities of the path. Therefore, we propose to use the minimum over all paths of the path slack divided by the number of activities on the path as a robustness measure,

$$RM_7(\sigma) = \min_{\pi} \left\{ \frac{s_\pi^\sigma}{|\pi|} \right\}. \tag{8}$$

Although the number of paths can be exponentially large, we can evaluate this robustness measure efficiently by computing for each $k = 1$ to $n + 2$ the shortest path in the schedule with exactly k activities.

In many cases, a reasonable estimate of the variance of the uncertainty can be made as well, even if the exact distribution of the uncertainty is unknown. We can exploit this additional information by making the assumption that the duration of each activity is normally distributed, as normal distributions can be characterized solely by their mean and variance. Although this assumption might be wrong for the distribution of the duration of a single activity, if follows from the central limit theorem that the sum of activity durations does resemble a normal distribution. Therefore, we can approximate the uncertainty in the duration of a path in the schedule.

We can utilize this approximation as the basis for several robustness measures. Firstly, we propose another path-based robustness measure. Analog to the minimum weighted path slack in RM_7, we use the minimum probability that a path can be completed within the deadline, computed over all the possible paths in the graph. That is, we compute for each path π the normal distribution approximation X_π of the duration of the path by summing the processing time distributions of activities on that path, and report the minimum probability of completion before the deadline:

$$RM_8(\sigma) = \min_{\pi} \{P(X_\pi \leq T)\} \tag{9}$$

Although the paths in the schedule are connected by precedence relations, they are assumed to be independent by this robustness measure.

In contrast to RM_7, we might have to evaluate all the paths in the schedule to compute the distribution-based robustness measure RM_8, since the usual graph-theoretical properties of paths, such as the property that any sub-path of a shortest path is again a shortest path, do not hold in this case. To keep the computation tractable, we construct in topological order for each activity i the set of paths in schedule σ ending at i, from the paths ending at the immediate predecessors of i:

$$\Pi_i = \left\{ (\pi^1, \ldots, \pi^k, i) \mid \exists\, j \prec i \in \mathcal{POS}_\sigma : (\pi^1, \ldots, \pi^k) \in \Pi_j \right\}. \tag{10}$$

Furthermore, we compute the distribution X_π of the duration of each $\pi \in \Pi_i$ by summing the normal distributions of the activities on the path. We can then reformulate RM_8 to

$$RM_8(\sigma) = \min_{\pi \in \Pi_{n+1}} P(X_\pi \leq T). \tag{11}$$

To avoid the exponential growth of Π_i, we repeatedly remove the path π from Π_i with the smallest probability of exceeding any other path in Π_i, until the set of paths is at most size K. Ties are broken randomly in this pruning procedure, and a maximum size of $K = 8$ was shown to be sufficiently large to achieve a good robustness estimation in preliminary experiments.

Another approach is to estimate the distribution of total makespan of the schedule. Many approximation algorithms have been proposed in literature, see [9] for a comparison of several techniques. An efficient method to construct an approximation of the makespan distribution is to evaluate the activities in topological order, computing the makespan distribution Y_i up to each activity i as the distribution of the maximum over the makespan distributions of the immediate predecessors of i

$$Y_i = \max_{j \prec i \in \mathcal{POS}_\sigma} \{Y_j\} + D_i, \tag{12}$$

where D_i is the normal approximation of the activity duration of i, and the maximum over the predecessor distributions is approximated with a normal distribution as proposed by [11]. The robustness measure, which is proposed in the work of [12], is then

$$RM_9(\sigma) = P\left(Y_{n+1} \leq T\right). \tag{13}$$

5 Experimental setup

The main application of surrogate robustness measures is in the comparison of schedules, since these can estimate the true robustness of a schedule far more efficiently than other approaches such as simulation. However, we need to investigate whether the estimations correctly reflect the relative ordering of schedules according to their robustness to verify that the robustness estimators are actually suitable for this purpose. To accomplish this, we construct empirical makespan distributions of a set of realistic schedules in a simulation study, and search for robustness estimators that show a strong correlation with the empirical results.

We have selected two real-world instances of the shunting problem described in Section 3 as the basis of our simulation study. The first one originates from *"Kleine Binkchorst (KBH)"*, which is a shunting yard of the NS near the central station of The Hague. It consists of a single night during which 19 train units arrive at the yard. These train units need to receive internal cleaning and a maintenance inspection; three of them need to be washed as well. Due to all the necessary train movements, shunting plans of problem instance typically have close to 160 activities, with 250 to 300 precedence relations. The other instance is obtained from a shunting yard near Utrecht, named *"OZ"*, which contains, contrary to the KBH, many dead-end tracks. As a result, the main difficulty in the scheduling problem is the parking order of the trains. This instance has 16 train units and a total of 27 service activities. The number of activities in the corresponding shunting plans ranges from 140 to 160 activities, and roughly 300 precedence relations.

For each of these two scheduling problems, we generated 500 feasible shunting plans with a local search algorithm that, starting with an infeasible initial solution, iteratively alters the current solution to resolve conflicts in the shunting plan. The meta-heuristic used in the local search framework is *simulated annealing*, which is a stochastic optimization technique that accepts with a small probability some deteriorations in the solution quality – the number of conflicts – due to local modifications. Initial solutions are generated by scheduling the service activities in a random order and assigning the trains to random parking tracks. Furthermore,

the objective that is optimized by the local search consists only of the sum of the (weighted) conflicts, and the search process stops when a feasible shunting plan is found. Due to the stochastic nature of the solution method, this generation process produces a diverse set of shunting plans. See [13] for more details of the local search algorithm.

The main components of the uncertainty in the execution of a shunting plan are the arrival time of trains and the duration of service activities and train movements. Disturbances in the arrival time of a train are modeled with a uniform distribution with the mean equal to the scheduled arrival time, and an interval size of 10 minutes. The service activities and train movements always have a nonnegative duration, and the size of the disruptions are usually proportional to the duration of the activities. Therefore, we model the uncertainty in these activities with log-normal distribution with the nominal duration as the mean, and a standard deviation equal to 0.1 times the nominal duration. Robustness measures RM_1 to RM_7 use only the nominal durations in their computation, while RM_8 and RM_9 require the standard deviation of the distributions as well. Although for RM_4, the slack sufficiency measure, we can pick any value between zero and one for the fraction λ, we set it equal to the standard deviation of the uncertainty of the service and movement activities, i.e., $\lambda = 0.1$, as is suggested in [7].

The schedules are then evaluated by each of the surrogate robustness measures listed in Section 4 to generate their predictions of the robustness of the schedules. The predictions are compared with the results of a Monte Carlo simulation, which is a technique that repeatedly draws samples from the distribution of the uncertainty – thus simulating different realizations of the scheduling problem – to approximate the makespan distribution of the schedule. To obtain an accurate empirical distribution of the makespan, we collect 20000 samples per schedule.

Since the objective of the shunting yard planners at the NS is to find feasible shunting plans that minimize the probability of delayed departures, we use the fraction of samples in which the schedule realization resulted in the delay as the primary performance metric. Additionally, we compute the average lateness of the empirical makespan distribution to get a better understanding of the problem structure.

The correlation between the performance metrics and the robustness measures is investigated in the following section by computing both the *Pearson correlation coefficient* (r), and *Spearman's rank correlation coefficient* (ρ). If the robustness of a schedule can be approximated by a robustness measure, then a high value of the measure should indicate a low delayed fraction and average lateness, and we expect that the robustness measure will have a correlation coefficient close to -1 with either of the performance metrics. A coefficient of -1 for the Pearson correlation means that there is a perfect linear relation between the robustness measure and the performance metric, and a robustness measure with a Spearman correlation of -1 will rank the schedules perfectly according to the performance metric. The Spearman correlation coefficient is particularly suitable for our experiments, since the purpose of the robustness measures is to compare schedules efficiently.

In addition to the two robustness performance metrics, we record the time required by each robustness measure to evaluate the schedules to compare the computational efficiency of the measures. To obtain reliable estimates of these computation times, we evaluated each of the 500 schedules 100 times with every robustness measure, and compute the average computation time per evaluation.

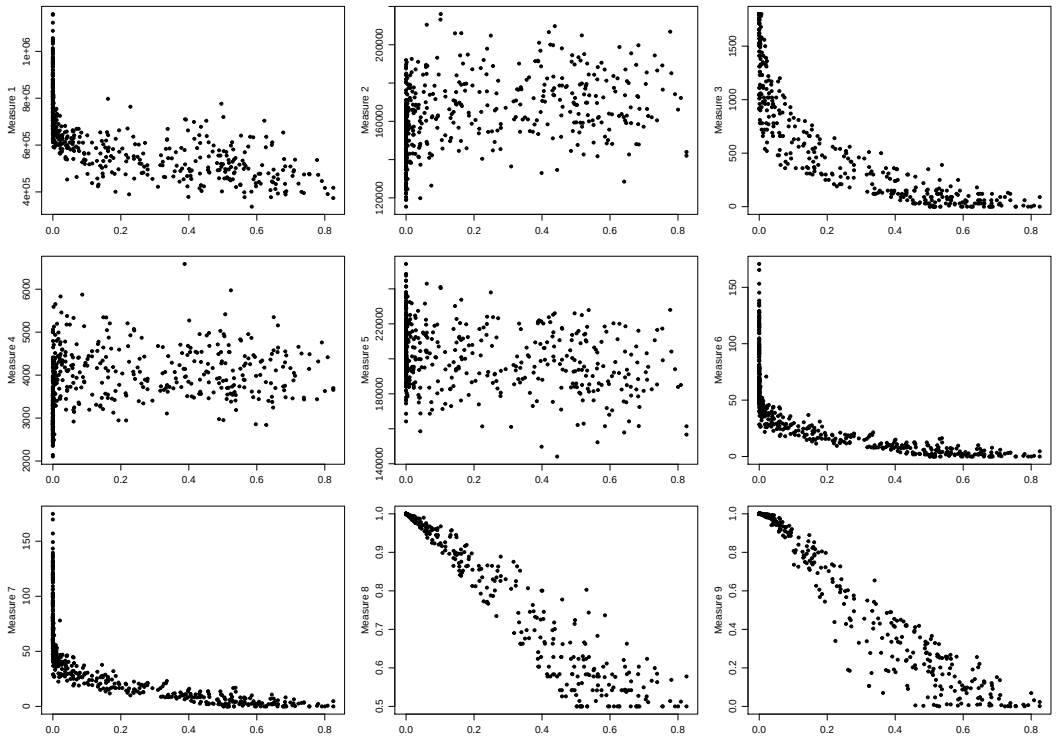

Figure 1 Scatter plots for the 9 robustness measures, showing the computed value of the measure (vertical axis) and the fraction of delayed samples (horizontal axis) of the KBH instances.

Table 1 The Spearman (ρ) and Pearson (r) correlation coefficients, as well as the average computation time, for the KBH instances. Coefficients close to -1 indicate that the robustness measure is a good approximation of the schedule robustness.

	Fraction delayed		**Average Lateness**		**Computation**
	ρ	r	ρ	r	**Time (ms)**
RM_1	-0.840	-0.663	-0.840	-0.828	0,01
RM_2	0.456	0.357	0.470	0.491	0,02
RM_3	-0.955	-0.838	-0.972	-0.990	0,01
RM_4	0.457	0.290	0.479	0.507	0,54
RM_5	-0.298	-0.293	-0.321	-0.326	3,95
RM_6	-0.964	-0.718	-0.955	-0.889	6,64
RM_7	-0.963	-0.719	-0.953	-0.887	0,87
RM_8	-0.982	-0.969	-0.972	-0.835	0,57
RM_9	-0.981	-0.971	-0.971	-0.846	0,23

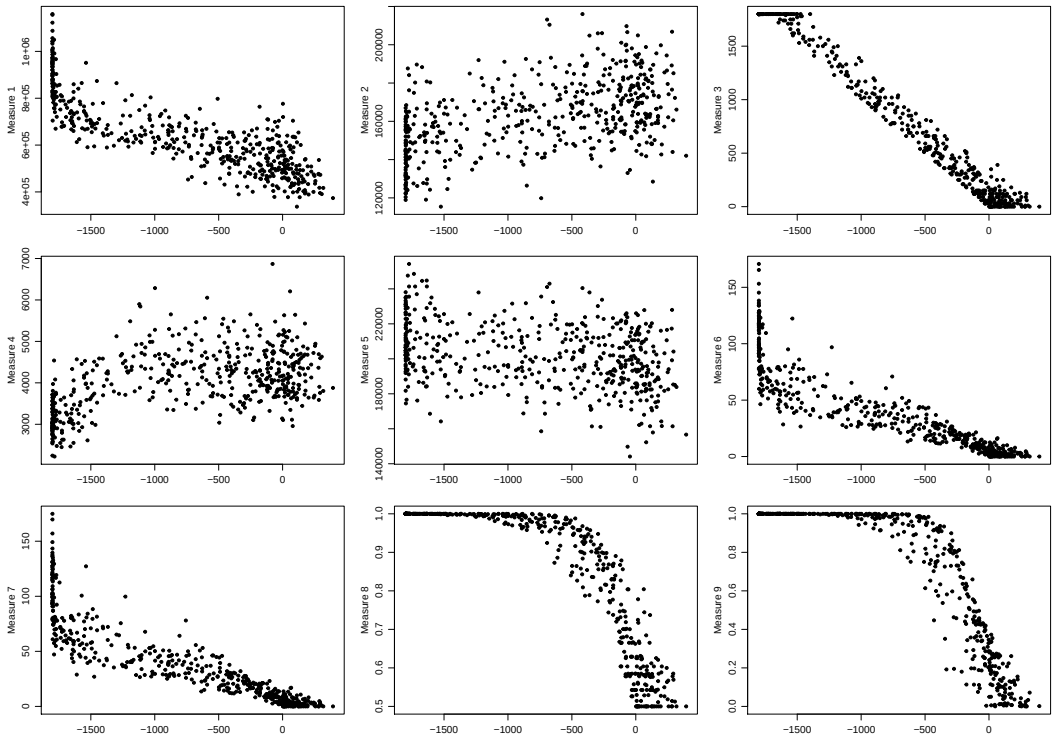

Figure 2 Scatter plots for the 9 robustness measures, showing the computed value of the measure (vertical axis) and the average lateness over the samples (horizontal axis) of the KBH instances.

Table 2 The Spearman (ρ) and Pearson (r) correlation coefficients, as well as the average computation time, for the OZ instances. Coefficients close to -1 indicate that the robustness measure is a good approximation of the schedule robustness.

	Fraction delayed		**Average Lateness**		**Computation**
	ρ	r	ρ	r	**Time (ms)**
RM_1	-0.933	-0.831	-0.943	-0.962	0,01
RM_2	0.544	0.510	0.542	0.606	0,01
RM_3	-0.975	-0.876	-0.980	-0.989	0,01
RM_4	0.156	0.132	0.161	0.270	0,53
RM_5	-0.118	-0.117	-0.127	-0.151	3,81
RM_6	-0.969	-0.840	-0.976	-0.972	6,28
RM_7	-0.968	-0.841	-0.975	-0.972	0,83
RM_8	-0.977	-0.959	-0.971	-0.818	0,60
RM_9	-0.972	-0.910	-0.960	-0.896	0,25

6 Empirical results

The relations between the robustness measures and the fraction of samples in which trains departed with a delay in the simulation study are shown in Figure 1 for the *Kleine Binckhorst* test set. Figure 2 shows the results for the average lateness performance metric of same instances. Tables 1 and 2 list the correlation coefficients of the robustness measures and the two performance metrics, as well as the average computation time, of the KBH and OZ instances.

The two robustness measures based on normal approximations, RM_8 and RM_9, appear to have the strongest rank correlation with both the performance metrics, clearly showing the advantage of exploiting the additional information of the variance of the uncertainty. Furthermore, both measures show a high Pearson correlation coefficient with the delayed fraction metric, as can be seen in Figure 1. If we take the computation time into account as well, then the approximation of makespan distribution, RM_9, would be the preferred robustness measure in a practical application.

When knowledge of the variance is not available, robustness measures that rely only on the nominal processing time of activities have to be used. Of those measures, RM_3, RM_6 and RM_7 are good choices in practice due to their high correlation with both performance metrics. In particular, the minimum total slack RM_3, which is equivalent to the makespan of a schedule, shows a remarkably strong Spearman correlation with the robustness performance metrics, and the correlation appears to be linear with the average lateness metric, which is an alternative formulation of the expected makespan. Given that the makespan can be computed more efficiently than the normal approximation methods, this robustness measure will most likely be sufficient to obtain robust solutions to scheduling problems with deadlines.

Contrary to the result of [7], the robustness measure RM_2, RM_4 and RM_5, which are based on maximizing the sum of the free slacks, correlate poorly to either of the performance metrics. This result is supported by the random scattering of the three measures in Figures 1 and 2. In the case of RM_2 and RM_4, the probability of delays in the schedule actually increases with the total amount of free slack in the schedule. Although the cause of this relation remains to be investigated, one possible explanation might be that free slack in these shunting plans mostly arises when a train is scheduled to wait until a route or a resource is available for its movement or service activity. Therefore, if a shunting plan contains many waiting trains, then the infrastructure or resources at the shunting yard are not used effectively, and the schedule will likely have a large makespan.

7 Conclusion

In this paper, we have studied robustness measures for shunting plans, which are solutions to the scheduling problem with deadlines that arises at shunting yards. The goal of the research is to identify measures that can accurately and efficiently estimate the robustness of a shunting plan, which is the likelihood that all trains depart on time from the shunting yard when disruptions occur in the operational phase. To achieve this goal, we have proposed new path-based robustness measures, and compared these, as well as several existing measures, with the results of a Monte Carlo simulation study on shunting plans for two real-world shunting problems of the Dutch Railways. We have shown that the new and existing robustness measures that utilize normally distributed approximations of the activity durations are strongly correlated with robustness of the schedules. Despite its simplicity, the makespan is also a good indicator of the robustness for schedules with deadlines. Contrary to earlier results on schedules without deadlines, the free slack has a poor predictive value of the robustness

of shunting plans. Further research should be conducted to investigate the differences in robustness in scheduling problems that are subject to deadlines, and those that require the minimization of the makespan.

References

1 Mohammad A. Al-Fawzan and Mohamed Haouari. A bi-objective model for robust resource-constrained project scheduling. *International Journal of production economics*, 96(2):175–187, 2005.

2 Louis-Claude Canon and Emmanuel Jeannot. Evaluation and optimization of the robustness of dag schedules in heterogeneous environments. *IEEE Transactions on Parallel and Distributed Systems*, 21(4):532–546, 2010.

3 Hédi Chtourou and Mohamed Haouari. A two-stage-priority-rule-based algorithm for robust resource-constrained project scheduling. *Computers & industrial engineering*, 55(1):183–194, 2008.

4 Rina Dechter, Itay Meiri, and Judea Pearl. Temporal constraint networks. *Artificial intelligence*, 49(1-3):61–95, 1991.

5 Öncü Hazır, Mohamed Haouari, and Erdal Erel. Robust scheduling and robustness measures for the discrete time/cost trade-off problem. *European Journal of Operational Research*, 207(2):633–643, 2010.

6 V. Jorge Leon, S. David Wu, and Robert H. Storer. Robustness measures and robust scheduling for job shops. *IIE transactions*, 26(5):32–43, 1994.

7 Mohamed Ali Khemakhem and Hédi Chtourou. Efficient robustness measures for the resource-constrained project scheduling problem. *International Journal of Industrial and Systems Engineering*, 14(2):245–267, 2013.

8 Przemysław Kobylański and Dorota Kuchta. A note on the paper by Ma Al-Fawzan and M. Haouari about a bi-objective problem for robust resource-constrained project scheduling. *International Journal of Production Economics*, 107(2):496–501, 2007.

9 Arfst Ludwig, Rolf H. Möhring, and Frederik Stork. A computational study on bounding the makespan distribution in stochastic project networks. *Annals of Operations Research*, 102(1-4):49–64, 2001.

10 Simon Mountakis, Tomas Klos, and Cees Witteveen. Temporal flexibility revisited: Maximizing flexibility by computing bipartite matchings. In *ICAPS*, pages 174–178, 2015.

11 Saralees Nadarajah and Samuel Kotz. Exact distribution of the max/min of two gaussian random variables. *IEEE Transactions on very large scale integration (VLSI) systems*, 16(2):210–212, 2008.

12 Guido Passage, Han Hoogeveen, and Marjan van den Akker. Combining local search and heuristics for solving robust parallel machine scheduling. Master's thesis, Utrecht University, 2016. https://dspace.library.uu.nl/bitstream/handle/1874/334269/thesis.pdf.

13 Roel van den Broek, Han Hoogeveen, Marjan van den Akker, and Bob Huisman. A local search algorithm for train unit shunting with service scheduling, 2018. Manuscript submitted for publication.

14 Michel Wilson. *Robust scheduling in an uncertain environment*. PhD thesis, TU Delft, 2016.

15 Michel Wilson, Tomas Klos, Cees Witteveen, and Bob Huisman. Flexibility and decoupling in simple temporal networks. *Artificial Intelligence*, 214:26–44, 2014.