Evaluating Repair Strategies for a Water-Treatment Facility using Arcade*

B.R. Haverkort[1,2] M. Kuntz[3] A. Remke[1] S. Roolvink[1] M.I.A. Stoelinga[1]
[1]University of Twente, NL, [2]Embedded Systems Institute, NL, [3]University of Konstanz, D

Abstract

The performance and dependability of critical infrastructures, such as water-treatment facilities is essential. In this paper we use various performance and dependability measures to analyze a simplified model of a water treatment facility. Building on the existing architectural framework Arcade a model is derived in XML format and then automatically mapped to the model checker PRISM. Using the stochastic model checking capabilities that PRISM offers, we compare different repair strategies, with respect to their costs, system reliability, availability and survivability. For this case study we conclude that using non-preemtive priority scheduling with additional repair crews is the best choice with respect to performance, dependability and costs.

1 Introduction

Over the last decade, we have seen an increased awareness in governments around the world about the vulnerability and dependence of society on so-called critical infrastructures. For instance, the Dutch government has recently identified 13 critical infrastructures [11]. Among these is the electricity supply infrastructure (the power distribution grid and power generation), which is the research topic of many projects. Another important critical infrastructure which is much less studied is water distribution and water treatment. In this paper, we focus on the last four phases of a water treatment facility; the key issue here is that water companies need to provide their service of delivering high-quality water at all times. A recent survey in the Netherlands found that water-treatment facilities, including their embedded Supervisory Control And Data Acquisition (SCADA) systems, are highly vulnerable [10] to failures and attacks. Therefore, the study of the impact of failures (whether physical failures or cyber attacks) is of vital importance.

To facilitate the analysis of performance and dependability requirements the dependability framework Arcade has been introduced [5] which can be linked - unambiguously - to existing design tools. Recently an XML-based input language [9] has been introduced for Arcade. It has a precise underlying semantics and, at the same time, can be coupled easily to both design tools and analysis tools. Hence, this approach pairs rigor with applicability and openness with respect to true design tools.

In contrast to earlier dependability analysis with Arcade, that analyzes reliability and availability only, here, we propose a new performance measure called *quantitative survivability* that is an refinement of survivability as defined by Cloth et al. [7]. Survivability is defined there as the probability of timely recovery after the occurrence of predefined disaster.

Throughout the paper, we analyze a simplified water treatment facility. For this model we compare different repair strategies with respect to their costs and their effect on system reliability, availability and qualitative and quantitative versions of survivability. Note that we are working with a Dutch water company on a much more detailed model, that however contains classified information.

The case study is modeled in terms of stochastic reactive modules [1] instead of Input/Output Interactive Markov chains(I/O-IMCs), as proposed in [5]. We implement a translation from Arcade - XML to PRISM, tailored to the need of our case study, instead of the mapping to CADP that has been proposed earlier. Our choice for PRISM has pragmatic reasons: to analyse the case study, we crucially need stochastic model checking, to express the various performability measures (survivability, repair cost, etc). Where as PRISM supports the stochastic model checking of CSL (continuous stochastic logic, [4]) and CSRL (Continuous Stochastic Reward Logic, [6][3]), CADP does not provide stochastic model checking capabilities.

There is a substantial existing body of work on the modeling and analysis of critical infrastructures; for an overview we refer the reader to [12]. However, most papers in the literature focus either on modeling, or combine modeling with simulation. In contrast, we combine modeling with exact analysis, based on stochastic model checking.

The remainder of the paper is organized as follows. We

*This research was funded by CTIT: Centre for Telematics and Information Technology Research institute and 3TU.CeDICT: Centre for Dependable ICT Systems.

introduce the new tool chain in Section 2. The relevant measures are described in Section 3. In Section 4 we present the watertreatment facility and in Section 5 we show its analysis results. Finally, Section 6 presents our conclusions.

2 The modeling and analysis framework

The architectural framework Arcade has been shown to be a very useful tool for modeling and analysis of performance and dependability measures [5]. The framework distinguishes three types of components: (1) Basic components, which describe the components in terms of their operational and failure behavior; (2) Repair units, which repair components under their responsibility. (3) Spare management units, which activate spare components when their primary is down. Inputs to the Arcade framework are: (1) an architectural model, given in an XML format (based on [9]), that describes the system in terms of basic components, repair strategies, and spare management units (2) a fault tree that describes when the system is down and (3) a dependability measure specification.

For this case study, we add costs to the model to express, for example, the price of repair and focus on performability measures like for example, survivability and incurred costs through repair. To enable the evaluation of various repair strategies w.r.t. their costs and the evaluation of quantitative and qualitative survivability analysis, stochastic model checking capabilities are needed. The original Arcade tool chain (using the CADP tool [8]) does not provide these. However, unlike CADP, PRISM supports the stochastic model checking of CSL (continuous stochastic logic, [4]) and CSRL (Continuous Stochastic Reward Logic, [6]), which are essential for comparing costs and for survivability analysis. Therefor, we choose to use Arcade in combination with PRISM. Fig. 1 presents an overview of our tool chain. Arcade translates its input to the Input/Output Interactive Markov chain model, whereas in this paper we translate to the input language of the PRISM model checker in CTMC mode. I/O-IMC are more expressive than CTMCs, and the I/O-IMC parallel composition differs semantically from those used in PRISM. Nondeterminism, present in I/O-IMC, is the distinguishing feature, and is needed for full Arcade. Nevertheless we made sure that the two translations agree – in the sense that they lead to identical results – for the constructs occuring in this case study. These constructs are basic components, and/or connectors, and different repair strategies: dedicated, first-come-first-serve, fastest repair first, fastest failure first. If two components have the same repair or failure rate, then we apply first-come-first-serve. Notably, failures are assured to never occur simultaneousy. Simultaneous failures are a notorious source of nondeterminism, so their absence is a prerequisite for applying the PRISM translation.

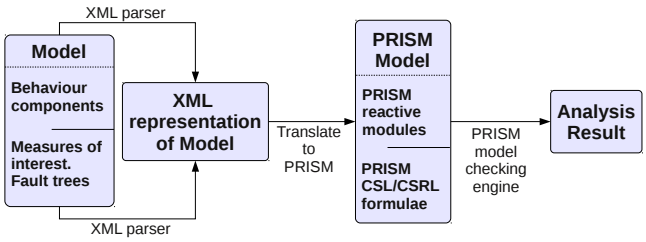

Figure 1. The tool chain via PRISM

3 Measures of interest

In this section we define the measures of interest using the PRISM CSL/CSRL notation. Arcade uses a fault tree to define when the system is down. This fault tree is an AND/OR expression whose literals represent the failure modes of the component [5].

Reliability, i.e. the continuity of correct service [2] is the probability of having no system failure within a certain mission time t. While the statespace S contains all the states, we define S_{down} as the set of states of S where the fault tree evaluates to 1. The set of operational states is defined as $S_{\text{operational}} = S \setminus S_{\text{down}}$. Reliability is then expressed as: $\mathcal{P}_{Reliability} = 1 - \mathcal{P}_{Unreliability}$, where $\mathcal{P}_{Unreliability} =?$ [true U<=t S_{down}].

Availability, i.e., the long run probability that the system is operational [2] assuming that components can be repaired: $\mathcal{S}_{Availibility} =? [S_{\text{operational}}]$.

Cloth et al.[7] define *survivability* as the ability of a system to recover to a predefined service level in a timely manner after the occurrence of disasters. Hence, we choose the initial distribution such that model starts in the disaster. Then, the probability that the system returns before time *t* to a state where the required service is provided is expressed as $\mathcal{P}_{Recovery} =?$ [true U<=t $S_{service}$]. Note that, $S_{service}$ is a set of states where the required service is provided.

In classical dependability analysis service and failures are expressed in a qualitative way. However, we introduce a quantitative measure of service, called service level, that can take a value from the interval $[0, 1]$ and describes a fraction of the maximum possible service.

To derive the quantitative service level, first the fault tree is converted into a quantitative service tree by substituting AND gates by OR gates and vice versa. Using the classical interpretation of the gates, the resulting service tree then evaluates to 1 if still some form of service is delivered and to 0 if the system is in a failure state. The quantitative interpretation of gates is as follows.

Consider a gate with inputs $x_1, x_2, \ldots, x_n$. The quantitative AND gate AND_q is the minimum of its inputs, i.e.,

$$\text{AND}_\text{q}(x_1, \ldots, x_n) = \min(x_1, \ldots, x_n), \qquad (1)$$

and the quantitative OR gate OR_q is the average of the in-

puts and represents the fraction of available service, i.e.,

$$\mathrm{OR_q}(x_1, \ldots, x_n) = \frac{\sum_{i=1}^{n} x_i}{n}. \quad (2)$$

The service tree joins components that are connected in series by an AND gate, as the failure of one of these components is sufficient to disable service in that series. When considering the quantitative interpretation, the component with the minimum service forms the bottleneck of the system and hence defines the service of the complete line. Redundant components are connected by an OR gate, because if one of these components remains working, service is still delivered. However, the quantitative interpretation reflects the service degradation that occurs if some of the components that contribute to the overall service fail.

To compute the probability of reaching at least a service level $x \in [0, 1]$ all states for which the quantitative service tree evaluates to at least x are added to the set $S_{sl(x)}$.

Repair costs can be analyzed using rewards.

Instantaneous Cost yield the costs at a particular time instant and can be expressed by the CSRL formula $\mathcal{R}_{instantaneous} =? [I = t]$.

Accumulated cost represent the total cost up to a given time bound t and can be expressed by the CSRL formula $\mathcal{R}_{Accumulated} =? [C <= t]$.

4 Water-treatment facility: Description

We evaluated the availability, reliability, survivability and repair costs of a water-treatment facility. Fig. 2 shows a schematic representation of the water-treatment model. The system consists of two independent process lines, each consisting of a set of softening tanks, sand filters, pumps and a reservoir. Softening tanks reduce the hardness of the water by crystallizing the calcium, magnesium and certain other elements. Sand filters remove the last remaining impurities in the water, resulting in drinking water ready for consumption. A reservoir is used for temporary storage to cope with fluctuating water demands throughout the day. The pumps transport the drinking water to the customer via a distribution network. *Line 1*, consists of three softening tanks (ST), three sand filters (SF), one reservoir (RES) and four pumps (PUMP). For normal service, three pumps need to be functioning and the fourth pump is a spare, indicated by (3+1). The softening tanks and slow-sand filters are redundant components, and to achieve normal service they all have to be operational.

Line 2 consists of three Softening tanks, two Sand filters, one Reservoir and three Pumps. In *Line 2*normal service can be provided by two Pumps; the third Pump is a spare. Again, the Softening tanks and Sand filters are redundant components.

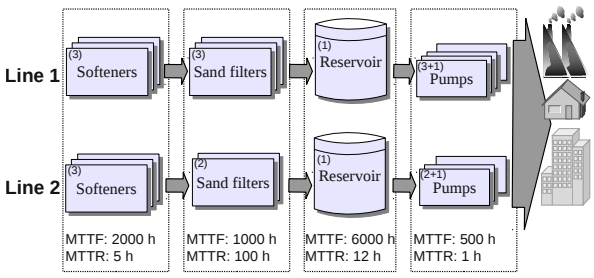

Figure 2. Water-treatment facility model

	Line 1		*Line 2*	
Strategy	**States**	**Trans.**	**States**	**Trans.**
Dedicated	2048	22528	512	4606
FRF-1	111809	388478	8129	25838
FRF-2	111809	500275	8129	33957
FFF-1	111809	367106	8129	23354
FFF-2	111809	478903	8129	31473

Table 1. State space for repair strategies.

In Fig. 2 the chosen Mean Time To Failure (MTTF) and Mean Time To Repair (MTTR) we use in our analysis are shown below the components. The real rates are not shown as they are classified. In our model all components can only fail in one mode, and have only a single operational mode.

In this paper we consider repair strategies with a single repair unit per line, that may contain several repair crews. We compare the following repair strategies: dedicated (DED), fastest repair first (FRF), fastest failure first (FFF), with either one or two repair crews.

5 Water-treatment facility: Evaluation

In this section we show the analisys results for the Water-treatment model as described in Section 4.

State space. Using the different repair strategies we get very different state space sizes as shown in Table 1. Each line of the system was seperately modelled to limit the state space for the analysis. The state space size for FRF and FFF does not change, if we have two instead of one repair crew, only the number of transitions increases. The reason for this is that the number of queue orders does not change, but the added repair crew does change the number of ways in which repairs are done.

Availability. The availability for *Line 1*, A_{line1}, and the availability for *Line 2*, A_{Line2}, are computed seperately. The overall availability is then given by: $A_{Line1} \cup A_{Line2} = A_{Line1} + A_{Line2} - A_{Line1}A_{Line2}$.

Table 2 shows the steady-state availability. Clearly dedicated (DED) repair provides the highest availability. The strategies with two repair crews yield just a slightly lower

Strategy	line 1	line 2	Combined
Dedicated	0.7442018	0.8186317	0.9536063
FRF-1	0.7225597	0.8101931	0.9473399
FRF-2	0.7439214	0.8186312	0.9535554
FFF-1	0.7273540	0.8120302	0.9487508
FFF-2	0.7440022	0.8186662	0.9535790

Table 2. Availability for repair strategies.

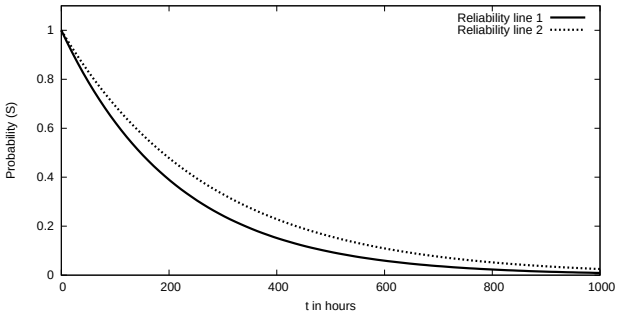

Figure 3. Reliability over time

availability, while one repair crew yields a significantly lower availability. Hence, we conclude that an additional repair crew greatly increases the availability of the system.

Reliability. This measure does not consider repairs, hence we do not distinguish between strategies. For the realiability we defined S_{down} as the set of states for which a process line is not fully operational. Note that in each lines one pump can fail for that line to be fully operational. In Fig. 3 the reliability of the water-treatment model is shown for both lines. Even though *Line 2* has less redundant components it is more reliable than *Line 1*. This is because: (1) the pumps have the shortest MTTF so they influence the reliability the most, (2) the probability that 2 pumps fail in *Line 1* is larger than for *Line 2* because, *Line 1* has four pumps that can fail whereas *Line 2* has only three, and (3) the other phases still operate with only one component;

Survivability and Costs. We analyzed the survivability of the water-treatment model after the occurrence of the following two disasters: (1) All pumps in the system fail, and (2) in *Line 2* two Pumps, one Softener, one Sand filter, and the Reservoir fail.

We analyze the survivability for all possible service level values of $x \in [0, 1]$ given a disaster. From the results, we conclude that for *Disaster 1* the survivability results for all repair strategies with one repair crew are the same and also the repair strategies with two repair crews have the same results. This is because for *Disaster 1* only one sort of component fails so, the repair order is the same for these repair strategies. Therefore, we only show the results for *Disaster 1* for FRF-1, FRF-2 and DED.

Because we consider Given Occurrence Of Disaster (GOOD) models to analyze survivability, we do not know

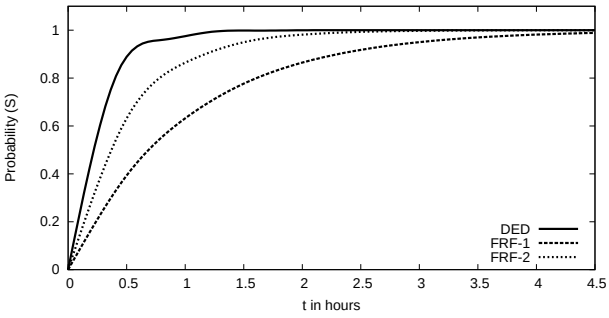

Figure 4. Survivability *Line 1*, *Disaster 1*, X_1

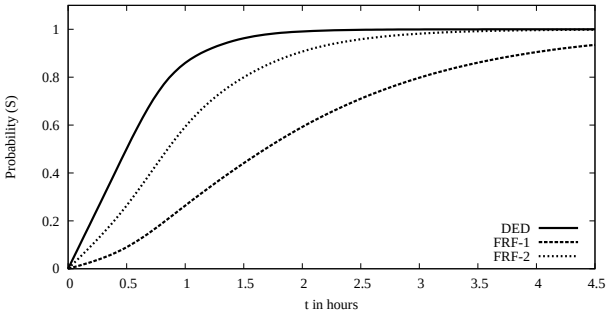

Figure 5. Survivability *Line 1*, *Disaster 1*, X_2

the order in which components have failed. However, this is important, as it defines the order in which components are repaired. Hence we use the priority of components to define the repair ordering.

For each of the repair strategies we compute the *Instantaneous* and *Accumulated* cost. For the accumulated cost start observing the cost directly after the occurence of the disaster. During normal operation the accumulated cost linearly increase according to the idle cost for the repair crews. The instantaneous cost of a system during normal operation is defined by the idle cost of the repair crews. After the occurrence of a disaster the system has an increased instantaneous cost which decreases and converges to the instantaneous cost for the system during normal operation. In the model each RU has an a cost of one per hour when idle and cost of zero when working. For a BC a cost of zero is applied when operational and three per hour when failed.

Line 1. From the numerical survivability results for *Line 1*[1], we conclude that there are three ranges for x of $S_{sl(x)}$ that give the same survivability results, namely: $X_1 = [0.33, 0.66)$, $X_2 = [0.66, 1)$ and $X_3 = [1, 1]$. The number of different service-intervals results from the amount of redundant componets in the different phases. Note, that spare components do not create extra service-intervals.

Fig. 4 and 5 show the recovery to respectively service-interval X_1 and X_2 after the occurrence of *Disaster 1* for the different repair strategies. In both figures we see that the extra repair crew in FRF-2 increases the recovery speed

[1]The numerical results are not shown in this paper.

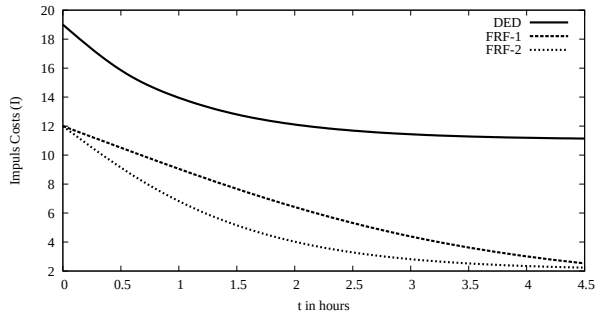

Figure 6. Instantaneous cost *Line 1*, *Disaster 1*

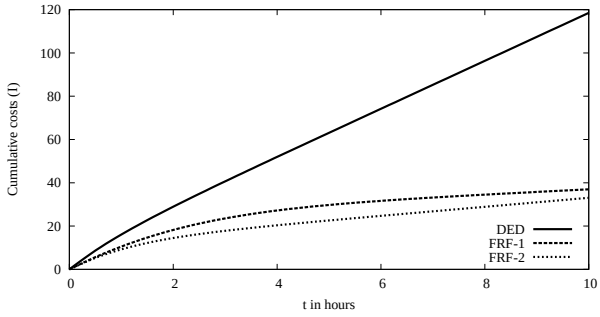

Figure 7. Accumulated cost *Line 1*, *Disaster 1*

compared to FRF-1. Also, DED has in both cases the fastest recovery. For service interval X_2 more components need to be repaired than for X_1. This corresponds to the slower recovery to X_2 compared to X_1.

To analyze the trade off between recovery and induced cost, Fig. 6 and 7 show the instantaneous and accumulated cost for the same scenario. Because DED has a repair crew for every component it always has the faster recovery, but it also has the highest instantaneous cost. The instantaneous cost for FRF-1 converges more slowly than FRF-2 and DED because, only one repair crew performs repairs. Hence, the number of components that need repairing decreases more slowly. For FRF-2 the instantaneous cost converges more slowly than DED which corresponds to the slower recovery speed of FRF-2 show in Fig. 4 and 5. As the instantaneous cost is the rate of increase of the accumulated cost we see that DED also has the highest accumulated cost. Because FRF-2 has, compared to FRF-1, a lower instantaneous cost during the recovery it also has a lower accumulated cost. We conclude that FRF-2 provides a good recovery after the occurrence *Disaster 1* while having a slightly lower cumulated cost than FRF-1 during the recovery. We also conclude that indeed DED provides that fastest recovery for the highest cost.

Line 2. In *Line 2* the combination of three redundant softeners with two redundant sandfilters and two redundant pumps results in four service-intervals. Again, the spare pump does not create extra service-intervals. We conclude that four ranges for x of $S_{sl(x)}$ give the same survivability results, namely: $X_1 = [0.33, 0.5)$, $X_2 = [0.5, 0.66)$, $X_3 = [0.66, 1)$ and $X_4 = [1, 1]$.

In Fig. 8 and 9 the recovery to service interval X_1 and X_3 given the occurrence of *Disaster 2* are shown. FFF-1 clearly provides the slowest recovery to X_1. This is because the Reservoir is repaired later in FFF-1 compared to FRF or DED, and without the reservoir no service is possible. While in service interval X_1 FFF-1 and FFF-2 provide a slower recovery than FRF-1 and FRF-2 in X_3 this is the other way. This is because for X_3 the sand filter becomes more important then the reservoir as it has a lower MTTF, a higher MTTR and also without the sand filter X_3 cannot be reached.

To analyze the trade off between recovery and cost, we show in Fig. 10 and 11 the instantaneous cost and the accumulated cost after the occurrence of *Disaster 2*. FFF-1 has the slowest convergence of the instantaneous cost, which corresponds to the slow recovery. For X_3 we see that FRF-1 has a slightly slower recovery than FFF-1 which should give FRF-1 a slower convergence of the instantaneous cost. However, because FFF-1 has more repeated failures of fast failing components (for example, a pump) it performs more repairs and thus has a slower decreasing instantaneous cost. When a component with a high MTTR is being repaired by FFF-1 these repeated failures will even increase the instantaneous cost as the number of failed components again increases this can be seen in Figure 10. Because of the slow instantaneous cost convergence of FFF-1 the accumulated cost is also the highest. From the figures we conclude Fastest Repair First with 2 repair crews is the best strategy as it has the fastest recovery to X_1 and the lowest accumulated cost.

For *Line 2* we conclude that DED provides the best survivability, but can only be used as a reference as it is unrealistic to use as it is to expensive and requires to many repair units. We also conclude that using Faster Repair First in combination with 2 repair crews is a very effective scheduling algorithm. However, we also see that the priority of components has a large impact on the recovery from a disaster. Analyzing survivability with respect to different service levels can help operators to select the ideal repair order. We can also conclude from these results that instantaneous costs can show the cost per repair strategy. Using accumulated cost shows which repair strategy gives a higher costs after a disaster. Using both survivability and costs will allow an operator to select a repair strategy that is both fast and inexpensive.

6 Conclusions

We have modeled a simplified water treatment facility using a subclass of the Arcade framework that has been mapped to PRISM reactive modules. PRISM then allows

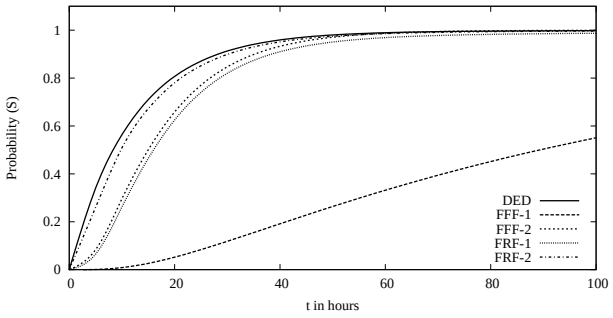

Figure 8. Survivability *Line 2*, *Disaster 2*, X_1

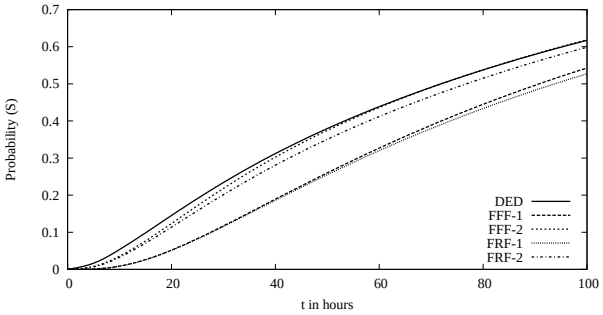

Figure 9. Survivability *Line 2*, *Disaster 2*, X_3

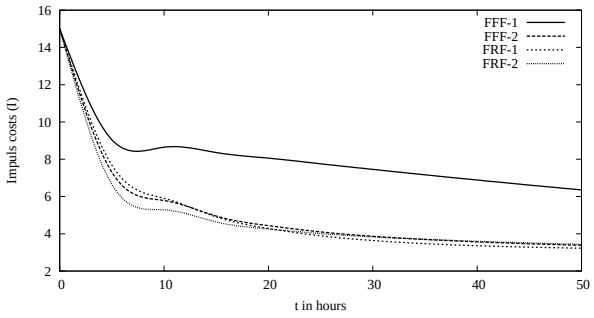

Figure 10. Instantaneous cost *Line 2 Disaster 2*

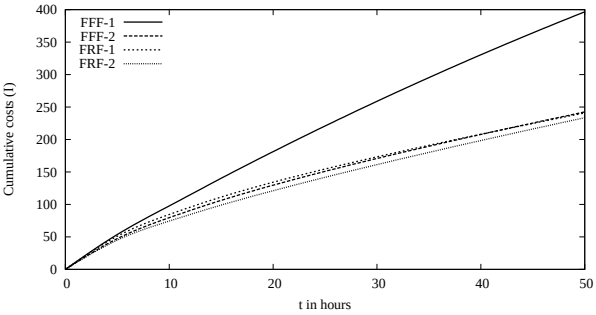

Figure 11. Accumulated cost *Line 2*, *Disaster 2*

to automatically derive various performability measures for the model including the introduced quantitative survivability. This leads us to the following conclusions. First, our analysis provides useful insights in the trade-offs between costs and reliability, availability and, in particular, quantitative survivability. Also, these performability measures are easy to express and compute using PRISM's CSL and CSRL model checking facilities. A problem we encountered is the fact that PRISM can only handle limited-size models. To deal with larger models, we plan to apply minimization techniques, which have shown to yield drastic reductions in the Arcade/CADP case [5].

Acknowledgment. We thank Pepijn Crouzen and Sascha Maass for their valuable help on the XML-format and Holger Hermans for the valuable discussions on the paper.

References

[1] R. Alur and T. Henzinger. Reactive Modules. *Formal Methods in System Design*, 15(1):7–48, 1999.

[2] A. Avizienis, J.-C. Laprie, B. Randell, and C. Landwehr. Basic Concepts and Taxonomy of Dependable and Secure Computing. *IEEE Transactions on Dependable and Secure Computing*, 1:11–33, 2004.

[3] C. Baier, L. Cloth, B. Haverkort, H. Hermans, and J. Katoen. Performability assessment by model checking of Markov reward models . *Formal Methods in System Design*, 35, 2010.

[4] C. Baier, B. Haverkort, H. Hermanns, and J. Katoen. Model-Checking Algorithms for Continuous-Time Markov Chains. *IEEE Transactions on Software Engeneering*, 29(7):1–18, July 2003.

[5] H. Boudali, P. Crouzen, B. R. Haverkort, M. Kuntz, and M. I. A. Stoelinga. Architectural dependability evaluation with Arcade. In *Proc. of DSN 2008*, pages 512–521, 2008.

[6] L. Cloth. *Model Checking Algorithms for Markov Reward Models*. PhD thesis, University of Twente, Enschede, Netherlands, 2006.

[7] L. Cloth and B. Haverkort. Model Checking for Survivability! In *Proc. of QEST 2005*, pages 145–154, 2005.

[8] H. Garavel, F. Lang, R. Mateescu, and W. Serwe. CADP 2006: A Toolbox for the Construction and Analysis of Distributed Processes. In *Proc. of CAV 2007*, volume 4590, pages 158–163. LNCS, 2007.

[9] H. Maass. Translating Arcade models into MoDeST code. Master's thesis, Saarland University, 2010. to appear.

[10] Ministerie van Binnenlandse Zaken en Koninkrijksrelaties. Bescherming Vitale Infrastructuur. Quick-scan naar vitale producten en diensten. Technical Report TNO FEL-03-C00, 2003.

[11] Ministerie van Binnenlandse Zaken en Koninkrijksrelaties. Raport bescherming vitale infrastructuur. Technical report, 2005. http://www.minbzk.nl/actueel /kamerstukken?ActItmIdt=54878.

[12] P. Pederson, D. Dudenhoeffer, S. Hartley, and M. Permann. Critical Infrastructure and Interdependency Modeling: A Survey of US and International Research. Technical Report INL/EXT-06-11464, Idaho National Laboratory, Department of Energy, 2006.