The Quantitative Verification Benchmark Set*

Arnd Hartmanns[1], Michaela Klauck[2], David Parker[3],
Tim Quatmann[4], and Enno Ruijters[1]

[1] University of Twente, Enschede, The Netherlands
[2] Saarland University, Saarbrücken, Germany
[3] University of Birmingham, Birmingham, United Kingdom
[4] RWTH Aachen, Aachen, Germany

Abstract. We present an extensive collection of quantitative models to facilitate the development, comparison, and benchmarking of new verification algorithms and tools. All models have a formal semantics in terms of extensions of Markov chains, are provided in the Jani format, and are documented by a comprehensive set of metadata. The collection is highly diverse: it includes established probabilistic verification and planning benchmarks, industrial case studies, models of biological systems, dynamic fault trees, and Petri net examples, all originally specified in a variety of modelling languages. It archives detailed tool performance data for each model, enabling immediate comparisons between tools and among tool versions over time. The collection is easy to access via a client-side web application at qcomp.org with powerful search and visualisation features. It can be extended via a Git-based submission process, and is openly accessible according to the terms of the CC-BY license.

1 Introduction

Quantitative verification is the analysis of formal models and requirements that capture probabilistic behaviour, hard and soft real-time aspects, or complex continuous dynamics. Its applications include probabilistic programs, safety-critical and fault-tolerant systems, biological processes, queueing systems, and planning in uncertain environments. Quantitative verification tools can, for example, compute the worst-case probability of failure within a time bound, the minimal expected cost to achieve a goal, or a Pareto-optimal control strategy balancing energy consumption versus the probability of unsafe behaviour. Two prominent such tools are Prism [15] for probabilistic and Uppaal [17] for real-time systems.

Over the past decade, various improvements and extensions have been made to quantitative model checking algorithms, with different approaches implemented in an increasing number of tools, e.g. [7,8,11,13,18]. Researchers, tool developers, non-academic users, and reviewers can all greatly benefit from a common set of realistic and challenging examples that new algorithms and tools are

* Authors are listed alphabetically. This work was supported by DFG grant 389792660 (part of CRC 248), ERC Advanced Grants 695614 (powver) and 781914 (frappant), NWO and BetterBe grant 628.010.006, and NWO VENI grant 639.021.754.

consistently benchmarked and compared on and that may indicate the practicality of a new method or tool. Such sets, and the associated push to standardised semantics, formats, and interfaces, have proven their usefulness in other areas such as software verification [4] and SMT solving [3].

In quantitative verification, the Prism Benchmark Suite (PBS) [16] has served this role for the past seven years. It provides 24 distinct examples in the Prism language covering discrete- and continuous time Markov chains (DTMC and CTMC), discrete-time Markov decision processes (MDP), and probabilistic timed automata (PTA). To date, it has been used in over 60 scientific papers. Yet several developments over the past seven years are not adequately reflected or supported by the PBS. New tools (1) support other modelling languages and semantics (in particular, several tools have converged on the Jani model exchange format [6]), and (2) exploit higher-level formalisms like Petri nets or fault trees. In addition, (3) today's quantitative verification tools employ a wide range of techniques, whereas the majority of models in the PBS work best with Prism's original BDD-based approach. Furthermore, (4) probabilistic verification and planning have been connected (e.g. [14]), and (5) MDP have gained in prominence through recent breakthroughs in AI and learning.

We present the Quantitative Verification Benchmark Set (QVBS): a new and growing collection of currently 72 models (Sect. 2) in the Jani format, documented by comprehensive metadata. It includes all models from the PBS plus a variety of new examples originally specified in significantly different modelling languages. It also covers decision processes in continuous stochastic time via Markov automata (MA [9]). The QVBS aggregates performance results obtained by different tools on its models (Sect. 3). All data is accessible via a client-side web application with powerful search and visualisation capabilities (Sect. 4).

2 A Collection of Quantitative Models

The Quantitative Verification Benchmark Set is characterised by commonality and diversity. All models are available in the Jani model exchange format [6], and they all have a well-defined formal semantics in terms of five related automata-based probabilistic models based on Markov chains. At the same time, the models of the QVBS originate from a number of different application domains, were specified in six modelling languages (with the original models plus information on the Jani conversion process being preserved in the QVBS), and pose different challenges including state space explosion, numeric difficulties, and rare events.

Syntax and semantics. The QVBS accepts any interesting model with a Jani translation to the DTMC, CTMC, MDP, MA, and PTA model types. Its current models were originally specified in Galileo for fault trees [20], GreatSPN [2] for Petri nets, the Modest language [5], PGCL for probabilistic programs [10], PPDDL for planning domains [21], and the Prism language [15]. By also storing the original model, structural information (such as in Petri nets or fault trees) that is lost by a conversion to an automata-based model is preserved for tools that

Table 1. Sources and domains of models

		source			application domain					
	all	PBS	IPPC	TA	com	rda	dpe	pso	bio	sec
all	72	24	10	7	12	9	17	16	6	5
DTMC	9	7			2	3	1			2
CTMC	13	7					4	1	6	
MDP	25	5	10		5	5		13		
MA	18			7		1	12	2		1
PTA	7	5			5					2

Table 2. Properties and valuations

		properties					parameter valuations				
	all	P	Pb	E	Eb	S	all	10^4	10^6	10^7	$>10^7$
all	229	90	57	52	12	18	589	135	127	94	28
DTMC	20	10	1	9			91	40	23	14	14
CTMC	49	6	22	4	11	6	161	43	52	28	5
MDP	61	40	3	17	1		82	31	24	21	6
MA	61	14	18	17		12	218	7	28	26	3
PTA	38	20	13	5			37	14		5	

can exploit it. We plan to broaden the scope to e.g. stochastic timed automata [5] or stochastic hybrid systems [1] in coordination with interested tool authors.

Sources and application domains. 41 of the QVBS's current 72 models stem from existing smaller and more specialised collections: 24 from the PRISM Benchmark Suite (PBS) [16], 10 from the probabilistic/uncertainty tracks of the 2006 and 2008 International Planning Competitions (IPPC) [21], and 7 repairable dynamic fault trees from the Twente Arberretum (TA) [19]. 65 of the models can be categorised as representing systems from six broad application domains: models of communication protocols (com), of more abstract randomised and distributed algorithms (rda), for dependability and performance evaluation (dpe), of planning, scheduling and operations management scenarios (pso), of biological processes (bio), and of mechanisms for security and privacy (sec). We summarise the sources and application domains of the QVBS models in Table 1.

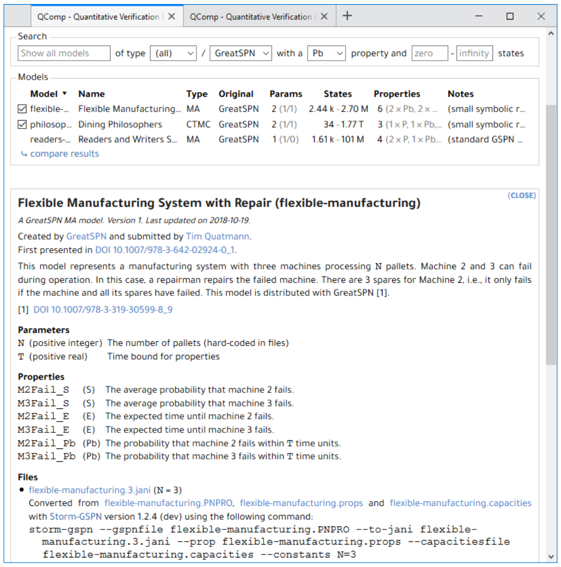
Metadata. Alongside each model, in original and JANI format, we store a comprehensive set of structured JSON metadata to facilitate browsing and data mining the benchmark set. This includes basic information such as a description of the model, its version history, and references to the original source and relevant literature. Almost all models are parameterised such that the difficulty of analysing the model can be varied: some parameters influence the size of the state spaces, others may be time bounds used in properties, etc. The metadata documents all parameters and the ranges of admissible values. It includes sets of "proposed" parameter valuations with corresponding state space sizes and reference results. Each model contains a set of properties to be analysed; they are categorised into probabilistic unbounded and bounded reachability (P and Pb), unbounded and bounded expected rewards (E and Eb), and steady-state queries (S). Table 2 summarises the number of properties of each type (left), and the number of suggested parameter valuations (right) per resulting state space size (if available), where e.g. column "10^6" lists the numbers of valuations yielding 10^4 to 10^6 states.

3 An Archive of Results

The Quantitative Verification Benchmark Set collects not only models, but also *results*: the values of the properties that have been checked and performance data on runtime and memory usage. For every model, we archive results obtained with different tools/tool versions and settings on different hardware in a structured Json format. The aim is to collect a "big dataset" of performance information that can be mined for patterns over tools, models, and time. It also gives developers of new tools and algorithms a quick indication of the relative performance of their implementation, saving the often cumbersome process of installing and running many third-party tools locally. Developers of existing tools may profit from an archive of the performance of their own tool, helping to highlight performance improvements—or pinpoint regressions—over time. The QVBS includes a graphical interface to aggregate and visualise this data (see Sect. 4 below).

4 Accessing the Benchmark Set

The models and results data of the Quantitative Verification Benchmark Set are managed in a Git repository at github.com/ahartmanns/qcomp. A user-friendly interface is provided at qcomp.org/benchmarks via a web application that dynamically loads the Json data and presents it in two views:


QComp - Quantitative Verification
QComp - Quantitative Verification
+
(CLOSE)

Search

Show all models of type (all) / GreatSPN with a Pb property and zero - infinity states

Models

	Model ▾	Name	Type	Original	Params	States	Properties	Notes
☑	flexible-...	Flexible Manufacturing...	MA	GreatSPN	2 (1/1)	2.44 k - 2.70 M	6 (2 × Pb, 2 × ...	(small symbolic r...
☑	philosop...	Dining Philosophers	CTMC	GreatSPN	2 (1/1)	34 - 1.77 T	3 (1 × P, 1 × Pb,...	(small symbolic r...
	readers-...	Readers and Writers S...	MA	GreatSPN	1 (1/0)	1.61 k - 101 M	4 (2 × P, 1 × Pb...	(standard GSPN ...

↳ compare results

Flexible Manufacturing System with Repair (flexible-manufacturing)

A GreatSPN MA model. Version 1. Last updated on 2018-10-19.

Created by GreatSPN and submitted by Tim Quatmann.
First presented in DOI 10.1007/978-3-642-02924-0_1.

This model represents a manufacturing system with three machines processing N pallets. Machine 2 and 3 can fail during operation. In this case, a repairman repairs the failed machine. There are 3 spares for Machine 2, i.e., it only fails if the machine and all its spares have failed. This model is distributed with GreatSPN [1].

[1] DOI 10.1007/978-3-319-30599-8_9

Parameters

N	(positive integer)	The number of pallets (hard-coded in files)
T	(positive real)	Time bound for properties

Properties

M2Fail_S	(S)	The average probability that machine 2 fails.
M3Fail_S	(S)	The average probability that machine 3 fails.
M2Fail_E	(E)	The expected time until machine 2 fails.
M3Fail_E	(E)	The expected time until machine 3 fails.
M2Fail_Pb	(Pb)	The probability that machine 2 fails within T time units.
M3Fail_Pb	(Pb)	The probability that machine 3 fails within T time units.

Files

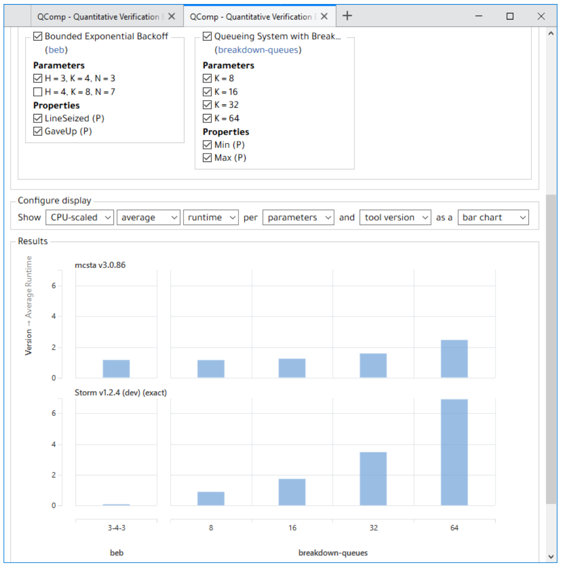
- flexible-manufacturing.3.jani (N = 3)
  Converted from flexible-manufacturing.PNPRO, flexible-manufacturing.props and flexible-manufacturing.capacities with Storm-GSPN version 1.2.4 (dev) using the following command:

```
storm-gspn --gspnfile flexible-manufacturing.PNPRO --to-jani flexible-
  manufacturing.3.jani --prop flexible-manufacturing.props --capacitiesfile
  flexible-manufacturing.capacities --constants N=3
```

Fig. 1. The model browser and detail view

The model browser presents a list of all models with key metadata. The list can be refined by a full-text search over the models' names, descriptions and notes, and by filters for model type, original modelling language, property types, and state space size. For example, a user could request the list of all Modest MDP models with an expected-reward property and at least ten million states. Every model can be opened in a detail view that links to the Jani and original files, shows all metadata including parameters, proposed valuations, and properties with reference results, and provides access to all archived results. Fig. 1 shows the model browser filtered to GreatSPN models that include a bounded probabilistic reachability property. The flexible-manufacturing model is open in detail view.

The results browser is accessed by selecting one or more models in the model browser and opening the "compare results" link. It provides a flexible, summarising view of the performance data collected from all archived results for the selected models. The data can be filtered to include select properties or parameter valuations only. It is visualised as a table or different types of charts, including bar charts and scatter plots. Fig. 2 shows the result browser for the beb and breakdown-queues models, comparing the performance of MCSTA [13] with default settings to Storm [8] in its slower "exact" mode. The performance data can optionally be normalised by the benchmark scores of the CPU used to somewhat improve comparability, although this still disregards many other important factors (like memory bandwidth and storage latency), of course.


QComp - Quantitative Verification
QComp - Quantitative Verification
+

☑ Bounded Exponential Backoff (beb)

Parameters
- ☑ H = 3, K = 4, N = 3
- □ H = 4, K = 8, N = 7

Properties
- ☑ LineSeized (P)
- ☑ GaveUp (P)

☑ Queueing System with Break... (breakdown-queues)

Parameters
- ☑ K = 8
- ☑ K = 16
- ☑ K = 32
- ☑ K = 64

Properties
- ☑ Min (P)
- ☑ Max (P)

Configure display

Show CPU-scaled average runtime per parameters and tool version as a bar chart

Results

mcsta v3.0.86
Storm v1.2.4 (dev) (exact)
Version → Average Runtime
0
2
4
6
0
2
4
6
3-4-3
8
16
32
64
beb
breakdown-queues

Fig. 2. The results browser showing a bar chart

The web application is entirely client-side: all data is loaded into the user's browser as needed. All aggregation, filtering, and visualisation is implemented in Javascript. The application thus has no requirements on the server side. It is part of the Git repository and can be downloaded and opened offline by anyone.

5 Conclusion

Building upon the successful foundation of the Prism Benchmark Suite, the new Quantitative Verification Benchmark Set not only expands the number and diversity of easily accessible benchmarks, but also professionalises the collection and provision of benchmark data through its Json-based formats for metadata and results. We expect its associated web application to become a valuable tool for researchers, tool authors, and users alike. The QVBS is also an *open* dataset: all content is available under the CC-BY license, and new content—new models, updates, and results—can be contributed via a well-defined Git-based process. The Quantitative Verification Benchmark Set is the sole source of models for QComp 2019 [12], the first friendly competition of quantitative verification tools.

Acknowledgments. The authors thank Paul Gainer (University of Liverpool), Sebastian Junges (RWTH Aachen), Joachim Klein (Technische Universität Dresden), Matthias Volk (RWTH Aachen), and Zhen Zhang (Utah State University) for submitting models to the QVBS, Gethin Norman (University of Glasgow) for his contributions to the Prism Benchmark Suite, and Marcel Steinmetz (Saarland University) for translating the IPPC benchmarks.

References

1. Abate, A., Blom, H., Cauchi, N., Haesaert, S., Hartmanns, A., Lesser, K., Oishi, M., Sivaramakrishnan, V., Soudjani, S., Vasile, C.I., Vinod, A.P.: ARCH-COMP18 category report: Stochastic modelling. In: ARCH Workshop at ADHS. EPiC Series in Computing, vol. 54, pp. 71–103. EasyChair (2018)
2. Amparore, E.G., Balbo, G., Beccuti, M., Donatelli, S., Franceschinis, G.: 30 years of GreatSPN. In: Principles of Performance and Reliability Modeling and Evaluation. pp. 227–254. Springer (2016)
3. Barrett, C., Fontaine, P., Tinelli, C.: SMT-LIB benchmarks, `http://smtlib.cs.uiowa.edu/benchmarks.shtml`
4. Beyer, D.: Software verification with validation of results (report on SV-COMP 2017). In: TACAS. LNCS, vol. 10206, pp. 331–349. Springer (2017)
5. Bohnenkamp, H.C., D'Argenio, P.R., Hermanns, H., Katoen, J.P.: MoDeST: A compositional modeling formalism for hard and softly timed systems. IEEE Trans. Software Eng. 32(10), 812–830 (2006)
6. Budde, C.E., Dehnert, C., Hahn, E.M., Hartmanns, A., Junges, S., Turrini, A.: JANI: Quantitative model and tool interaction. In: TACAS. LNCS, vol. 10206, pp. 151–168 (2017)
7. David, A., Jensen, P.G., Larsen, K.G., Mikucionis, M., Taankvist, J.H.: Uppaal Stratego. In: TACAS. LNCS, vol. 9035, pp. 206–211. Springer (2015)
8. Dehnert, C., Junges, S., Katoen, J.P., Volk, M.: A Storm is coming: A modern probabilistic model checker. In: CAV. LNCS, vol. 10427. Springer (2017)
9. Eisentraut, C., Hermanns, H., Zhang, L.: On probabilistic automata in continuous time. In: LICS. pp. 342–351. IEEE Computer Society (2010)
10. Gordon, A.D., Henzinger, T.A., Nori, A.V., Rajamani, S.K.: Probabilistic programming. In: FOSE. pp. 167–181. ACM (2014)
11. Hahn, E.M., Li, Y., Schewe, S., Turrini, A., Zhang, L.: iscasMc: A web-based probabilistic model checker. In: FM. LNCS, vol. 8442, pp. 312–317. Springer (2014)
12. Hartmanns, A., Hensel, C., Klauck, M., Klein, J., Kretínský, J., Parker, D., Quatmann, T., Ruijters, E., Steinmetz, M.: The 2019 comparison of tools for the analysis of quantitative formal models. In: TACAS. LNCS, vol. 11429. Springer (2019)
13. Hartmanns, A., Hermanns, H.: The Modest Toolset: An integrated environment for quantitative modelling and verification. In: TACAS. LNCS, vol. 8413, pp. 593–598. Springer (2014)
14. Klauck, M., Steinmetz, M., Hoffmann, J., Hermanns, H.: Compiling probabilistic model checking into probabilistic planning. In: ICAPS. AAAI Press (2018)
15. Kwiatkowska, M.Z., Norman, G., Parker, D.: PRISM 4.0: Verification of probabilistic real-time systems. In: CAV. LNCS, vol. 6806, pp. 585–591. Springer (2011)
16. Kwiatkowska, M.Z., Norman, G., Parker, D.: The PRISM benchmark suite. In: QEST. pp. 203–204. IEEE Computer Society (2012)
17. Larsen, K.G., Lorber, F., Nielsen, B.: 20 years of UPPAAL enabled industrial model-based validation and beyond. In: ISoLA. LNCS, vol. 11247. Springer (2018)
18. Legay, A., Sedwards, S., Traonouez, L.M.: Plasma Lab: A modular statistical model checking platform. In: ISoLA. LNCS, vol. 9952, pp. 77–93. Springer (2016)
19. Ruijters, E., Budde, C.E., C. Nakhaee, M., Stoelinga, M.I.A., Bucur, D., Hiemstra, D., Schivo, S.: The Twente Arberretum, `https://dftbenchmarks.utwente.nl/`
20. Sullivan, K.J., Dugan, J.B., Coppit, D.: The Galileo fault tree analysis tool. In: FTCS-29. pp. 232–235. IEEE Computer Society (1999)
21. Younes, H.L.S., Littman, M.L., Weissman, D., Asmuth, J.: The first probabilistic track of the Int. Planning Competition. J. Artif. Intell. Res. 24, 851–887 (2005)