Kanban Production System

AVACS S3 Benchmark*

[1] Albert-Ludwigs-Universität Freiburg, Fahnenbergplatz, 79085 Freiburg, Germany
[2] Carl von Ossietzky Universität Oldenburg, 26111 Oldenburg, Germany
[3] Universität des Saarlandes, 66041 Saarbrücken, Germany

1 Purpose of the Benchmark

We have developed a symbolic algorithm [4,6,5] for the computation of observational bisimulations on labeled transition systems like

- strong
- orthogonal
- branching
- delay
- η
- weak
- progressing
- safety

bisimulation. Although we have mainly developed the new algorithm for our STATEMATE models, we wanted to evaluate it on more standard domains like process-algebraic descriptions of state spaces. One of the classical examples is the description of a Kanban production system [1] which has already been used as case study by the model checker PRISM [3] and the symbolic performance evaluation tool CASPA [2].

2 Description of the Kanban Production System

The process-algebraic description of the Kanban system we used consists of four parallel processes that are synchronized among each other. The system itself can be parameterized by a maximum number of *workpieces* that each process can handle.

We performed two series of experiments with one adapted configuration of the Kanban system: To enable the sensible application of different kinds of bisimulation, we had to introduce unobservable transitions. We have done this by hiding all internal process actions, such that only the synchronization actions were visible. This kind of configuration could be of interest when inter-process communication only is analyzed.

The input file for our configuration is given in Figure 1. Please note that the variable *ini* denotes the number of workpieces along which the descriptions are parameterized.

* `http://www.avacs.org`

```
int ini = 6;

kanban := (hide tredo1, tok1, tback1, tredo2, tok2, tback2,
                tok3, tback3, tredo3, tredo4, tok4, tback4 in
          (k1(ini,0,0,0) |[tsync1_23]|
          (k2(ini,0,0,0) |[tsync1_23,tsync23_4]|
           k3(ini,0,0,0))) |[tsync23_4]|
           k4(ini,0,0,0)
          )

k1(a [ini],x [ini],y [ini],z [ini]) :=  [a>0] -> (in1,1) ; k1(a-1,x+1,y,z)
                                        [x>0] -> (tredo1,1) ; k1(a,x-1,y+1,z)
                                        [x>0] -> (tok1,1) ; k1(a,x-1,y,z+1)
                                        [y>0] -> (tback1,1) ; k1(a,x+1,y-1,z)
                                        [z>0] -> (tsync1_23,1) ; k1(a+1,x,y,z-1)

k2(w [ini],x [ini],y [ini],z [ini]) :=  [w>0] -> (tsync1_23,1) ; k2(w-1,x+1,y,z)
                                        [x>0] -> (tredo2,1) ; k2(w,x-1,y+1,z)
                                        [x>0] -> (tok2,1) ; k2(w,x-1,y,z+1)
                                        [y>0] -> (tback2,1) ; k2(w,x+1,y-1,z)
                                        [z>0] -> (tsync23_4,1) ; k2(w+1,x,y,z-1)

k3(w [ini],x [ini],y [ini],z [ini]) :=  [w>0] -> (tsync1_23,1) ; k3(w-1,x+1,y,z)
                                        [x>0] -> (tredo3,1) ; k3(w,x-1,y+1,z)
                                        [x>0] -> (tok3,1) ; k3(w,x-1,y,z+1)
                                        [y>0] -> (tback3,1) ; k3(w,x+1,y-1,z)
                                        [z>0] -> (tsync23_4,1) ; k3(w+1,x,y,z-1)

k4(w [ini],x [ini],y [ini],z [ini]) :=  [w>0] -> (tsync23_4,1) ; k4(w-1,x+1,y,z)
                                        [x>0] -> (tredo4,1) ; k4(w,x-1,y+1,z)
                                        [x>0] -> (tok4,1) ; k4(w,x-1,y,z+1)
                                        [y>0] -> (tback4,1) ; k4(w,x+1,y-1,z)
                                        [z>0] -> (tout4,1) ; k4(w+1,x,y,z-1)
```

Fig. 1. CASPA input for our configuration of the Kanban system.

3 Experimental Results

For generating symbolic LTS representations, we have applied the stochastic process algebra tool CASPA (see [2]) from which we extracted symbolic, i. e. BDD, representations for the Kanban system [1]. We applied our bisimulation tool SIGREF to the benchmarks and computed the minimal observationally equivalent system w. r. t. each of the bisimulation mentioned above.

The sizes of the inputs are displayed in table 1, the results of the minimization in table 2. All experiments were performed on a Dual Core AMD Opteron processor with 2.4 GHz clock frequency and 4 GB of main memory.

References

1. Gianfranco Ciardo and Marco Tilgner. On the use of Kronecker operators for the solution of generalized stochastic Petri nets. Technical Report 96-35, Institute for Computer Applications in Science and Engineering, 1996.

Table 1. Size of the benchmarks

Input		original	strong	orthogonal	branch.	delay	η	weak	progress.	safety
kanban-4	$\|S\|$	1.60e07	778485	51805	2785	2785	2785	2785	8964	2785
	$\|T\|$	7.44e07	6419550	327079	10932	10932	10932	10932	48755	10932
kanban-5	$\|S\|$	1.68e07	ML	228385	7366	7366	7366	7366	24643	7366
	$\|T\|$	1.34e08	ML	1577123	31795	31795	31795	31795	144881	31795
kanban-6	$\|S\|$	2.65e08	ML	TL	17010	17010	17010	17010	58463	17010
	$\|T\|$	1.69e09	ML	TL	78584	78584	78584	78584	363384	78584
kanban-7	$\|S\|$	2.68e08	ML	TL	35456	35456	35456	35456	124311	35456
	$\|T\|$	2.62e09	ML	TL	172382	172382	172382	172382	805684	172382
kanban-8	$\|S\|$	4.22e09	ML	TL	68217	68217	68217	68217	242858	68217
	$\|T\|$	2.91e10	ML	TL	345128	345128	345128	345128	1626231	345128
kanban-9	$\|S\|$	4.29e09	ML	TL	123070	123070	123070	123070	443479	123070
	$\|T\|$	4.11e10	ML	TL	642837	642837	642837	642837	3048435	642837

Table 2. Runtimes in seconds for the minimization of the Kanban models

Benchmark	strong	ortho.	branch.	delay	η	weak	progress.	safety
kanban-4	2915.40	2054.10	14.46	14.24	16.46	16.92	25.58	2.83
kanban-5	ML	33684.30	68.12	74.24	100.38	80.30	142.11	10.66
kanban-6	ML	TL	308.57	342.44	455.03	440.04	657.21	48.29
kanban-7	ML	TL	1137.11	1301.01	1781.65	1327.28	3775.63	218.17
kanban-8	ML	TL	3723.02	4033.71	5841.12	4544.03	7422.98	832.78
kanban-9	ML	TL	12869.90	15324.80	16793.90	12285.60	36913.80	3529.68

2. Matthias Kuntz, Markus Siegle, and Edith Werner. Symbolic performance and dependability evaluation with the tool CASPA. In *Proc. of FORTE*, volume 3236 of *LNCS*, pages 293–307, 2004.
3. M. Kwiatkowska, G. Norman, and D. Parker. Probabilistic symbolic model checking with PRISM: A hybrid approach. *J. STTT*, 6(2):128–142, 2004.
4. Ralf Wimmer, Marc Herbstritt, and Bernd Becker. Minimization of large state spaces using symbolic branching bisimulation. In Matteo Sonza Reorda, Ondřej Novák, Bernd Staube, Hana Kubátová, Zdeněk Kotásek, Pavel Kubalík, Raimund Ubar, and Jiří Bucek, editors, *Proceedings of the 9th IEEE Workshop on Design & Diagnostics of Electronic Circuits & Systems (DDECS)*, pages 9–14, Prague, Czech Republic, April 2006. IEEE Computer Society Press.
5. Ralf Wimmer, Marc Herbstritt, and Bernd Becker. Optimization techniques for BDD-based bisimulation minimization. In Hai Zhou and Enrico Macii, editors, *Proceedings of the 17th ACM Great Lakes Symposium on VLSI*, pages 405–410, Stresa, Italy, March 2007. ACM Press.
6. Ralf Wimmer, Marc Herbstritt, Holger Hermanns, Kelley Strampp, and Bernd Becker. Sigref – a symbolic bisimulation tool box. In Susanne Graf and Wenhui Zhang, editors, *Proceedings of the 4th International Symposium on Automated Technology for Verification and Analysis (ATVA)*, volume 4218 of *Lecture Notes in Computer Science*, pages 477–492, Beijing, China, October 2006. Springer-Verlag.