Randomised Consensus Protocol (NFM 2011)

AVACS S3
Phase 2

July 28, 2011

1 Description of the Benchmark

In this case study we applied *PARAM* 2.0α [2] on a randomised shared coin protocol by Aspnes and Herlihy [1], based on an existing *PRISM* model [3]. The setting consists of N processes sharing a counter c, which initially has the value 0. In addition, a value K is fixed for the protocol. Each process i decides to either decrement the counter with probability p_i or to increment it with probability $1 - p_i$. In contrast to the original model, we do not fix the p_i to $\frac{1}{2}$, but use them as parameters. After writing the counter, the process reads the value again and checks whether $c \leq -K \cdot N$ or $c \geq K \cdot N$. In the first case, the process votes 1, in the second it votes 2. In both cases, the process stops afterwards. If neither of the two cases hold, the process continues its execution. As all processes which have not yet voted try to access the counter at the same time, there is a nondeterministic choice on the access order.

Based on this setting, we asked for two properties. First, we were interested in solving a probabilistic formula. We asked whether for each execution of the protocol the probability that all processes finally terminate with a vote of 2 is at least $\frac{K-1}{2 \cdot K}$. With appropriate atomic propositions *finished* and *allCoinsEqualTwo*, this property can be expressed as $P_{\geq \frac{K-1}{2 \cdot K}}(true \ \cup \ (finished \ \wedge \ allCoinsEqualTwo))$.
Second, we examined a reward formula and asked whether the expected number of steps until all processes have voted is above 25, expressed as $R_{>25}(F \ finished)$.

2 Results

2.1 Probabilistic Formula

The results of *PARAM* for the formula $P_{\geq \frac{K-1}{2 \cdot K}}(true \cup (finished \wedge allCoinsEqualTwo))$ with parameters $N = 2$ and $K = 2$ is given in Figure 1.

The leftmost part of the figure provides the minimal probabilites among all schedulers that all processes terminate with a vote of 2, depending on the parameters p_i. With decreasing p_i, the probability that all processes vote 2 increases, since it becomes more likely that a process increases the counter. Thus, also the chance that finally $c \geq K \times N$

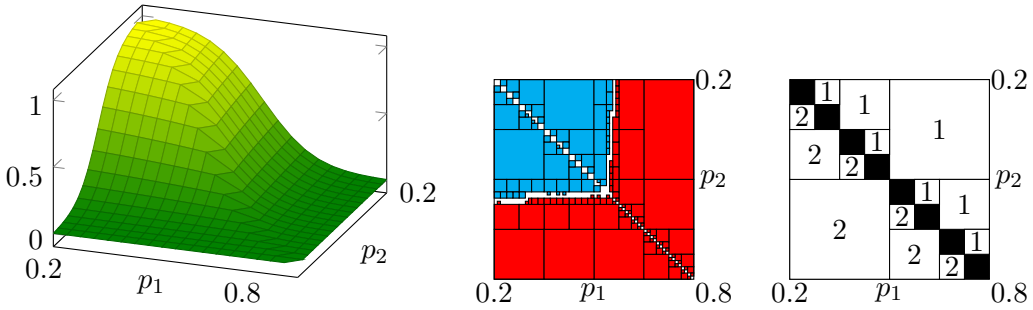

Figure 1: Results for the probabilistic formula with parameters $N = 2$ and $K = 2$

holds. The plot is symmetric, because both processes are independent and have an identical structure.

On the right part of the figure, we give an overview which schedulers are optimal for which parameter values. Here, boxes labelled with the same number share the same minimising scheduler. To obtain the minimal probability in case $p_1 < p_2$, the nondeterminism must be resolved such that the first process is activated if it has not yet voted. Doing so maximises the probability that we have $c \leq -K \cdot N$ before $c \geq K \cdot N$, and in turn minimises the probability that both processes vote 2. For $p_1 > p2$, the second process must be preferred.

In the middle part of the figure, we give the truth values of the formula. Cyan boxes correspond to regions where the property holds whereas red boxes depict regions where it does not hold. In white areas, the truth values is undecided. To keep the those areas viewable, we chose a rather high tolerance of 0.15. However, the property indeed holds in white areas enclosed by cyan areas and does not hold in white areas enclosed by red areas. The reason that these areas remain undecided is that the minimising scheduler changes at the diagonals, as discussed in the previous paragraph. If the optimal scheduler in a box is not constant for the region considered, we have to split it. Because the optimal scheduler always changes at the diagonals, some white boxes always remain.

2.2 Reward Formula

The results for the analysis of the formula $R_{>25}(F\ finished)$ are depicted in Figure 2. The leftmost part of the figure provides the expected number of steps for probabilities p_1 and p_2. The highest value is at $p_i = \frac{1}{2}$. Intuitively, in this case the counter does not have a tendency of drifting to either side and is likely to stay near 0 for a longer time. Again, white boxes surrounded by boxes of the same colour are those regions in which the minimising scheduler is constant. We see from the right part of the figure that this happens along four axes. For some values of the parameters, the minimising scheduler is not necessarily the one which always prioritises one of the processes. Instead, it may be necessary to schedule the first process, then the second again, etc. As we can see in the right part of Figure 2, this leads to a number of eight different schedulers to be considered for the considered variable ranges.

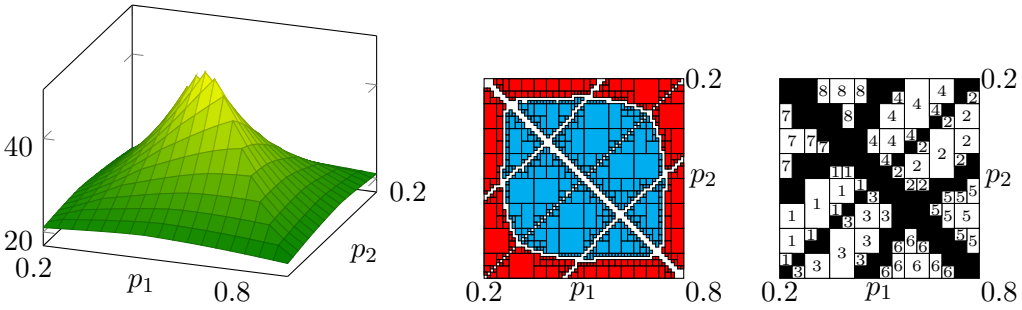

Figure 2: Results for the reward formula

2.3 Runtime

The runtime of our tool was measured on an Intel Core 2 Duo P9600 with 2.66 GHz running on Linux for both of the previously explained formulae. Thus, we considered two processes and different constants K. The results are given in Table 1. Column "States"

K	States	Until		Reward	
		min	truth	min	truth
2	272	4.7	22.8	278.8	944.7
3	400	13.7	56.7	4610.1	-
4	528	31.7	116.1	-	-
5	656	65.5	215.2	-	-
6	784	123.4	374.6	-	-
7	912	272.6	657.4	-	-

Table 1: Performance Statistics for the randomised consensus protocol

contains the number of states. The columns labelled with "Until" contain results of the first property while those labelled with "Reward" contain those of the second. Columns labelled with "min" contain just the time to compute the minimal values whereas those labelled with "truth" also include the time to compare this value against the bound of the formula. For all analyses, we chose a tolerance of $\epsilon = 0.05$. The time is given in seconds and "-" indicates that the analyses did not terminate within 90 minutes.

References

[1] James Aspnes and Maurice Herlihy. Fast randomized consensus using shared memory. *Journal of Algorithms*, 11(3):441–461, 1990.

[2] Ernst Moritz Hahn, Tingting Han, and Lijun Zhang. Synthesis for PCTL in Parametric Markov Decision Processes. In *NFM*, LNCS. Springer, 2011.

[3] M. Kwiatkowska, G. Norman, and R. Segala. Automated verification of a randomized distributed consensus protocol using Cadence SMV and PRISM. In *CAV*, volume 2102 of *LNCS*, pages 194–206. Springer, 2001.