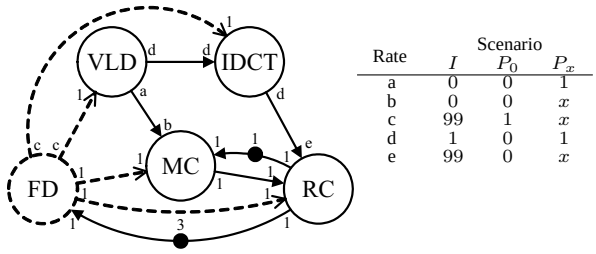
A Performance Analysis Tool for Scenario-Aware Streaming Applications

B.D. Theelen (B.D.Theelen@tue.nl)
Eindhoven University of Technology, Department of Electrical Engineering
P.O. Box 513, 5600 MB Eindhoven, The Netherlands

Abstract

Dataflow models are often used for analysing streaming applications. The recently introduced scenario-aware extension of the synchronous dataflow model can capture the dynamism in computation and communication resource requirements of streaming applications that originates from different modes of operation (scenarios). This scenario-aware dataflow model uses a probabilistic approach to express the order in which scenarios (and different execution times within a scenario) occur. This paper discusses a tool for exhaustive and simulation-based analysis of various important performance metrics for scenario-aware streaming applications.



Rate	Scenario I	P_0	P_x
a	0	0	1
b	0	0	x
c	99	1	x
d	1	0	1
e	99	0	x

Figure 1. SADF model of an MPEG-4 decoder.

1. Introduction

Streaming applications are commonly specified as a set of tasks, actors or processes with data and control dependencies to allow exploiting the parallel and pipelined execution capabilities of hardware platforms. The dependencies between processes can be cyclic, which indicates that different invocations of processes depend on each others results. The Synchronous Dataflow (SDF) model [5] (also known as Weighted Marked Graphs in Petri Net theory) can succinctly express these features. It also allows for design time analysis of many correctness and performance properties such as absence of deadlock and throughput [6]. However, SDF lacks support for expressing any form of dynamism. Such dynamism originates from distinct modes of operation or *scenarios* (like processing I, P or B frames in MPEG-4) in which a process may have substantially different resource requirements than in other operation modes [3]. Cyclo-Static Dataflow (CSDF) [1] is an extension of SDF that allows expressing cyclic patterns in the execution times of processes and/or the amount of data communicated between processes. In case probabilistic information is available on the order in which scenarios occur (which may not necessarily imply fixed reoccurring patterns), then the Scenario-Aware Dataflow (SADF) model [9] can be used. Despite the improved expressive power compared to SDF and CSDF, design-time analysis of correctness and performance properties remains possible. The advantage of using SADF instead of SDF or CSDF is therefore the potential of obtaining more realistic (performance) analysis results.

This paper discusses a performance analysis tool for streaming applications modelled with SADF. This tool is implemented as a module that extends SDF^3 [7] and allows exact analysis of various extrema, reachability and long-run performance metrics. It also offers simulation-based analysis as an alternative when state-space explosion issues render exact analysis infeasible. In that case, statistical information is provided on the accuracy of estimation results obtained for long-run performance metrics. The SADF Analysis Tool is freely available from `www.es.ele.tue.nl/sadf`.

2. Scenario-Aware Dataflow

Figure 1 shows (part of) the SADF model of an MPEG-4 decoder for the Simple Profile from [9], which supports video streams consisting of I and P frames. Such frames include a number of macro blocks (up to 99 for QCIF), each requiring operations like Variable Length Decoding (VLD), Inverse Discrete Cosine Transformation (IDCT), Motion Compensation (MC) and Reconstruction (RC). The model assumes an equal number (0 or $x \in \{30, 40, 50, 60, 70, 80, 99\}$, see [9]) of motion vectors and macro blocks to be decoded, each implying a different scenario. The vertices denote processes while the edges or *channels* represent (potential) dependencies. Two types of processes are distinguished. *Kernels* (solid vertices) represent the data processing part of a streaming application, whereas *detectors* (dashed vertices) model the control part that dynamically detects the scenarios. A *token* refers to a unit of information that is communicated between processes. The availability of tokens in the (in principle unbounded) FIFO buffer corresponding to a channel is shown with a dot. The 3 tokens in the channel from RC to FD express for example the ability to perform VLD and IDCT for macro blocks of 3 frames in a pipelined fashion. A *(production/consumption) rate* refers to the number of tokens that is produced/consumed by a process via a certain channel. In SADF, these rates can be 0 in certain scenarios to specify that data dependencies are absent or that kernels are inactive (like skipping VLD when decoding P frames for still video in scenario P_0). The Frame Detector (FD) detector in Figure 1 models that part of the actual VLD determining the frame type. It notifies all other processes about the detected frame type by sending control tokens via *control channels*. Such control channels are indicated with dashed arrows, while solid arrows denote ordinary *(data) channels*. When FD executes or *fires*, it determines the scenario in which VLD, ICDT, MC, RC and FD itself will operate by making a transition in a Markov chain that is associated with it (not shown). This results in fixing the value for the parameterised rate c and upon firing completion, FD sends control tokens valued with the detected scenario to VLD, IDCT, MC and RC. These kernels fix rates a, b, d and e by interpreting the control tokens before processing any data tokens. The execution time of a kernel (or detector) is determined based on an execution time distribution (with integer or real valued finite sample space) that is specified for each scenario in which it can operate. Notice that the scenario coherently affects the behaviour of for example the VLD and IDCT operations. Succinctly capturing such correlations that often exist between dynamic changes in resource requirements for different processes is a key feature of SADF.

3. Performance Analysis

Although the SADF Analysis Tool features amongst others an XML description format for specificating SADF models, visualisation of SADF models through the graph visualisation tool dotty [2] and functions to determine whether an SADF specification resembles in fact an SDF or CSDF model, the main functionality concerns the analysis of performance properties. Table 1 lists the metrics that can be evaluated using either state-space exploration based exact techniques or simulation-based estimation techniques by categorising them in extrema, reachability and long-run metrics.

The first metric in Table 1 refers to the maximum buffer occupancy of a channel that can occur for all possible (self-timed) schedules [6]. The expected response delay is an example of an expected reachability property as also considered in [4] and denotes the expected time until the first firing completion of a process. Throughput is defined in a similar way as for SDF and CSDF being the average number of firing completions of a process per time unit, which actually equals the reciprocal of the average time between two successive firing completions of that process (average inter-firing latency). Notice that the minimum/maximum as well as the variance in inter-firing latency give an indication for the variation in throughput. Two types of deadline miss probabilities can be analysed. The response delay deadline miss probability refers to the probability that the response delay for a process exceeds are certain deadline, whereas the periodic deadline miss probability indicates the probability that the inter-firing latency of a process exceeds such deadline requirement. The average and variance in buffer occupancy are defined as time average and variance respectively to compensate for the duration of each individual occupation that may occur [8]. These metrics allow analysing the utilisation of memory for the buffers compared to their maximum occupancy (which may assist in finding possibilities for sharing such memory).

The formal semantics defined in [9] makes SADF models amenable to rigorous analysis. Any SADF defines a *Timed Probabilistic System* (TPS) similarly as for models expressed with the Parallel Object-Oriented Specification Language [8, 10] or as a Probabilistic Timed Automata [4]. Such a TPS captures the non-deterministic choices originating from concurrency in an SADF and the probabilistic choices originating from the execution time distributions and Markov chains associated with detectors for determining scenarios. A TPS explicitly distinguishes performing action transitions like completing a firing from advancing time. After resolving non-determinism and shifting the information on the transitions into the states (which allows to define (temporal) reward functions on the states), the TPS reduces to a discrete-time Markov chain that is amenable to traditional techniques for exact and simulation-based performance analysis [8, 10]. Any SADF implies a TPS that is time and action determinate and additionally, the results for all metrics in Table 1 except for the maximum buffer occupancy metric[1] are invariant to the policy used for resolving non-determinism [9]. Exploiting these properties gives several opportunities for state-space reduction in case of exact analysis. However, traditional partial ordering reductions are insufficient to make analysis of for example the MPEG-4 decoder in Figure 1 feasible. The SADF Analysis Tool relies therefore also on an efficient implementation of the generally applicable state-space reduction technique introduced in [8]. Finally, we remark that computing the long-run metrics in Table 1 requires the involved SADF to be ergodic, which is verified (partly) based on the results in [9]. During the computation of any metric, the SADF Analysis Tool also checks whether the SADF is deadlock-free.

Metric	Type
Max Buffer Occupancy	Worst Case
Min/Max Response Delay	Best/Worst Case
Min/Max Inter-Firing Latency	Best/Worst Case
Response Delay Deadline Miss Probability	Probabilistic Reachability
Expected Response Delay	Expected Reachability
Throughput	Event Rate
Periodic Deadline Miss Probability	Sample Average
Average Inter-Firing Latency	Sample Average
Variance in Inter-Firing Latency	Sample Variance
Average Buffer Occupancy	Time Average
Variance in Buffer Occupancy	Time Variance

Table 1. Supported Performance Metrics

In case of simulation-based analysis, the results are determined based on the reward values observed for the path generated through the TPS. For long-run metrics, statistical information on the accuracy is given by determining confidence intervals using the algebra of confidence intervals from [8].

4. Results

Despite that storing the TPS implied by SADF models of realistic applications like the MPEG-4 decoder requires more than 3GB (even after resolving non-determinism), it is feasible to compute for example the throughput of RC (the output process). It takes 1449s on a P4 at 3Ghz to compute that the throughput equals 1.06378. This is much more realistic than the 0.43856 that would be obtained based on a corresponding SDF model. When considering a pipelining degree of only 2 instead of 3, computing the new reduced throughput of 1.05388 takes only 43s. Conversely, more complex examples have shown to suffer too much from state-space explosion, which is why simulation-based analysis is offered as an alternative. For a pipelining degree of 3, the simulation automatically terminates after 21s to obtain estimation results with relative errors below 5% for all long-run average metrics. This approach yields an estimated throughput of $1.05487 \pm 0.9\%$, which is very close to the computed throughput of 1.06378. Next to extending the SDF^3 tool for analysing SADF models and eliminating some restrictions on execution times for SDF, the SADF Analysis Tool is also a competitive alternative to SDF^3 for analysing SDF models in terms of run times.

References

[1] G. Bilsen, et al. Cyclo-Static Dataflow. *Transactions on Signal Processing*, vol 44, no 2, pp 397–408, IEEE, 1996.

[2] E. Gansner and S. North. An Open Graph Visualization System and its Applications to Software Engineering. *Software: Practice and Experience*, vol 30, no 11, pp 1203-1233, 2000.

[3] S.V. Gheorghita, T. Basten and H. Corporaal. Application Scenarios in Streaming-Oriented Embedded System Design. *Proceedings of SoC'06*, pp 175–178, IEEE, 2006.

[4] M. Kwiatkowska, et al. Performance Analysis of Probabilistic Timed Automata using Digital Clocks. *Formal Methods in System Design*, vol 29, pp 33–78, 2006.

[5] E. Lee and D. Messerschmitt. Synchronous Data Flow. *IEEE Proceedings*, vol 75, no 9, pp 1235–1245, 1987.

[6] S. Siram and S.S. Bhattacharyya. *Embedded Multiprocessors; Scheduling and Synchronization*. Marcel Dekker, 2000.

[7] S. Stuijk, M.C.W. Geilen and T. Basten. SDF^3: SDF for Free. *Proceedings of ACSD'06*, pp 276–278, IEEE, 2006.

[8] B.D. Theelen. *Performance Modelling for System-Level Design*. PhD Thesis, Eindhoven University of Technology, 2004.

[9] B.D. Theelen, et al. A Scenario-Aware Data Flow Model for Combined Long-Run Average and Worst-Case Performance Analysis. *Proceedings of MEMOCODE'06*, pp 185–194, IEEE, 2006.

[10] J.P.M. Voeten. Performance Evaluation with Temporal Rewards. *Performance Evaluation*, vol 50, no 2/3, pp 189–218, 2002.

1 Properly resolving non-determinism for the maximum buffer occupancy metric allows applying all the mentioned techniques in that case as well.