Data Structures – Third Year Report

LIF

May 2006

AMETIST DELIVERABLE 2.3.c

Project acronym: AMETIST
Project full title: Advanced Methods for Timed Systems
Project no.: IST-2001-35304

1 Introduction

This task 2.3 is dedicated to work on efficient data structures used for algorithms for the analysis of timed systems as part of AMETIST technology. The chain of such automated analysis consists typically of the following phases :

- modeling in a suitable modeling formalism,
- specification of the analysis goal,
- automatic coding to a more abstract search or constraint satisfaction problem,
- automatic analysis of the transformed problem.

From a theoretical point of view, many such automated analysis problems fall into complexity classes which are decidable yet NP-complete or PSPACE-complete. However, when we develop tools, we have to decide on an implementation technique that requires a software engineer's perspective : What are the concrete data structures to use for best performance. This choice will not change the worst case complexity of the algorithms, but can substantially influence the usability of the algorithms for the size and kind of problems we consider: Answers must be computed in an acceptable amount of time using a given amount of memory. The - contradictory - design goals to follow are thus :

- optimize data structures to allow faster computation;
- compact data structures so as either to fit bigger problems into the memory or to compute a result faster by a better memory usage.

The summary of the work done throughout the three years of AMETIST, concerning algorithms and data structures, is quite remarkable; a decent measurable progress that allows to deal with bigger problem instances today than before the start of the project is evident.

Concerning the state of the art before the project, we refer to the first year report.

2 Major contributions to this task in year 3

2.1 Difference bounds matrices and their applications

Difference bounds matrices (DBMs) are the essence of timed automata tools. They give a symbolic representation of sets of clock values. Naturally, in AMETIST we worked on this fundamental data structure throughout the project and also in the last year.

The significance of DBMs is underlined in the development of UppAal by separating from the main tool a library with an open interface for DBMs: This library represents now UppAal's constraint solving core, which can thus be used by other programs.

For verification purposes, one usually uses zone-based abstractions with respect to the maximal constants to which clocks of the timed automaton are compared. In [2, 3], it is shown that by distinguishing maximal lower and upper bounds, significantly coarser abstractions can be obtained, resulting in fewer symbolic states and consequently less memory and computation time. Soundness and completeness of the new abstractions with respect to reachability is shown and it is demonstrated how information about lower and upper bounds can be used to optimize the algorithm for transforming a difference bound matrix into normal form. The new techniques dramatically increase the scalability of UppAal for cases with asymetric bounds.

In another direction of research, in [7] an algorithm to compute efficient DBM substractions is studied. The substractions proposed guarantee a minimal number of splits and disjoint DBMs to avoid redundancy. Recall that a DBM represents a particular convex polyhedron. These polyhedra are closed under intersection, but not under union or complementation, which can only be represented by lists, i.e. unions, of DBMs. In particular, this means that the substraction of a DBM from another may result in a splitting into several DBMs and the splits may propagate. It is of prime importance to reduce the number of generated DBMs in an exploration loop, e.g., for reachability, because the result is propagated and serves to compute further successors later. Applications of DBM substraction concern priced timed automata [9] and timed games [4]. Figure 2.1 shows in the first three columns the benefit of the new technique.

Plates	Pre	Min+Disj	Disj	Min	None
4	20515	10M, 1.1s	10M, 1.5s	10M, 1.1s	10M, 1.2s
5	88678	43M, 7.4s	46M, 17.7s	43M, 8.0s	43M, 16.4s
6	454679	249M, 108.4s	400M, 18772s	249M, 110.7s	249M, 157.6s

Figure 1: Zone difference strategies in applications to timed games

An interesting application of DBMs is shown in [6], where DBMs are used in test generation for real time systems. The role of DBMs is to reduce the number of test cases by symbolic methods.

2.2 Distributed reachability analysis in timed automata

In [1], the problem of timed automata reachability analysis on clusters of PCs is studied. The aim of computing on clusters is, obviously, to combine the resources of machines in a network for more powerful compution for complex problems without too much communication and memory overhead. For timed automata reachability, this poses particular difficulties, as the data structures combine a mixture of hash tables and zone lists. Behrmann explores strategies for this and evaluates them with a distributed version of UppAal

2.3 Symmetry reduction

Symmetry reduction was known for untimed systems for a long time, but an adaptation to timed automata required a serious effort. The fundamental algorithms and data structures were developed in the first two years of AMETIST and the prototype exposed an enormous reduction potential, lifting the size of symmetric systems that can be handled by several orders of magnitude. From a data structure perspective, symmetry reduction requires a representation of symbolic states allowing for permutations and reductions to a normal form.

While no new fundamental results were obtained in the third year, symmetry reduction has been integrated into the distribution of UppAal, thus making the resulting speedup accessible to the user community.

2.4 Logic approaches

In the first two years, we worked on the reduction of timed automata reachability and related problems to a satisfaction solving problem of constraints that mix inequalities and boolean formulas. In [5], significant progress is reported on the side of the constraint solver, which now incorporates state of the art data structures known from discrete SAT solvers, and which applies particular strategies for the detection of unsatisfiable subsets of numerical constraints. It improves the performance of the previous solver by several orders of magnitude in some relevant classes of problems.

2.5 Trace based methods

The *event zone approach* as an alternative approach to classical DBMs for the symbolic exploration of timed automata has been developed from the beginning of the AMETIST project. It solves a particular aspect of the combinatory explosion occurring with classical DBMs, the splitting of symbolic states depending on the order of transition occurrences, even if these transitions concern unrelated components in a parallel system. The basic benefit from the method is that symbolic paths which differ only in the order of independent transitions lead to the same event zone, thus avoiding zone splitting. In the third year, in [8], we have simplified the presentation of the approach on the theoretical level.

Also in the third year, three important aspects were worked on that were, however, brought to conclusion only after the end of the project. We therefore report it in this deliverable as it emerged clearly from the AMETIST funding :

1. The incorporation of local state invariants.
2. A richer input language.
3. An identification of sequences that have been explored before up to commutation.

Concerning (2), the prototype tool "Else" has been completely rewritten and experimental results show that it can be significantly faster and more memory efficient than UppAal where zone splitting is a problem. Figure 2.5 shows the performance of the tool on a planning example compared to UppAal: The left column shows that without event zones (without reduction), it is significantly less efficient than UppAal, whereas with the reduction, it can outperform UppAal by several orders of magnitude.

Number of lanes	No partial order		With partial order		UppAal	
	time	memory	time	memory	time	memory
1	0.02s	16m	0.03s	16m	0.02s	6m
2	0.03s	16m	0.03s	16m	0.02s	6m
3	0.06s	16m	0.04s	16m	0.03s	6m
4	1.98s	22m	0.30s	17m	0.23s	7m
5	548.57s	279m	1.29s	19m	20.35s	29m
6	34681.91s(*)	2301m	10.80s	36m	2946.67s	438m
7	—	—	87.35s	119m	—	—
8	—	—	554.35s	466m	—	—

Figure 2: The event zone prototype compared to UppAal on a planning example

Concerning (3), an implementation of "Mazurkievicz Traces" has been realized that also improves the "Local First Search" method reported in the first year. For timed automata, while the number of symbolic states remains unchanged, we have observed speedups of up to a factor 4 by incorporating this trace library into the event zone approach (see Figure 2.5).

Number of philosophers	Memory	Time	
		Testing	No Testing
2	16m	0.03s	0.02s
3	16m	0.05s	0.05s
4	17m	0.31s	0.52s
5	22m	5.01s	9.58s
6	72m	78.12s	173.33s
7	540m	1168.92s	2840.52s

Number of lanes	Memory	Time	
		Testing	No Testing
1	16m	0.03s	0.03s
2	16m	0.03s	0.03s
3	16m	0.03s	0.04s
4	17m	0.09s	0.12s
5	19m	1.27s	2.31s
6	36m	10.79s	22.96s
7	118m	87.46s	211.78s
8	466m	554.34	1490.43s

Figure 3: Example speed ups of event zone exploration with trace library

3 Conclusion

The work done in this task has produced many interesting ideas and a significant practical progress in data structures and algorithms for timed automata - as was to be expected from the project.

This progress has been continuous throughout the three years of the project and beyond: The work on the tools and algorithms is pursued along with other research directions started during the AMETIST project.

The most significant data structure for timed automata, the DBM, was and is in the focus of interest:

- DBMs were adapted in the first and second year to include optimization concepts and to compete with state of the art scheduling algorithms.
- DBMs were extended for new applications requiring new operations.
- DBMs were improved by new abstraction techniques, symmetry and partial order concepts.

It has to be remarked that a lot of the work in this task is hidden in many person months of programming effort. Some of this work concerns locally small optimizations with significant effects contributing to the overall performance of tools. Such work is only documented in the source code of programs. However, some improvements and new uses of data structures with either outstanding effects or which require formal justification, result in published work. We concentrated on the latter improvements in this discussion.

In summary, this task is thus a fundamental contribution to the whole AMETIST project.

References

[1] G. Behrmann. Distributed reachability analysis in timed automata. *Software Tools for Technology Transfer*, 7(1):19–30, 2005. Available from World Wide Web: `http://www.cs.aau.dk/~kgl/AMETIST/Year3/behrmann05.ps`.

[2] G. Behrmann, P. Bouyer, K. G. Larsen, and R. Pelánek. Lower and upper bounds in zone based abstractions of timed automata. In *Proc. 10th Int. Conf. Tools and Algorithms for the Construction and Analysis of Systems (TACAS'2004)*, volume 2988 of *Lecture Notes in Computer Science*, pages 312–326. Springer-Verlag, 2004. Available from World Wide Web: `http://www.lsv.ens-cachan.fr/Publis/PAPERS/BBLP-tacas04.ps`.

[3] G. Behrmann, P. Bouyer, K.G. Larsen, and R. Pelanek. Zone based abstractions for timed automata exploiting lower and upper bounds. *Software Tools for Technology Transfer*, 2005. Available from World Wide Web: `http://www.cs.aau.dk/~kgl/AMETIST/Year3/bblp05.pdf`. Selected papers from TACAS'04.

[4] F. Cassez, A. David, E. Fleury, K.G. Larsen, and D. Lime. Efficient on-the-fly algorithms for the analysis of timed games. In Luca de Alfaro Martin Abadi, editor, *To appear in Proceedings of CONCUR 2005*, Lecture Notes in Computer Science. Springer Verlag, 2005. Available from World Wide Web: `http://www.cs.aau.dk/~kgl/AMETIST/Year3/cdfll05.ps`.

[5] S. Cotton, E. Asarin, O. Maler, and P. Niebert. Some progress in satisfiabilty checking for difference logic. In *FORMATS/FTRTFT'04*, number 3253 in LNCS, pages 263–276. Springer, 2004. Available from World Wide Web: `http://www-verimag.imag.fr/~maler/Papers/dlsat.ps`.

[6] M. Krichen and S. Tripakis. An expressive and implementable formal framework for testing real-time systems. In *The 17th IFIP Intl. Conf. on Testing of Communicating Systems TESTCOM'05)*, LNCS. Springer, 2005. To appear.

[7] Kim G. Larsen, Alexandre David, John Haakansson, and Paul Pettersson. Minimal dbm substraction. In Paul Pettersson and Wang Yi, editors, *Proceedings of 16th Nordic Workshop on Programming Theory*, number 2004-041 in Uppsala technical report, pages 17–21. Uppsala University, October 2004. Available from World Wide Web: `http://www.cs.aau.dk/~kgl/AMETIST/Year3/DBM.ps`.

[8] Denis Lugiez, Peter Niebert, and Sarah Zennou. A partial order semantics approach to the clock explosion problem of timed automata. In Kurt Jensen and Andreas Podelski, editors, *accepted for Theoretical Computer Science - special issue on selected papers of TACAS 2004*, volume 2988 of *LNCS*, page 40 pages. Springer-Verlag, 2005. Available from World Wide Web: `http://www.cmi.univ-mrs.fr/~niebert/docs/specialissue.pdf`.

[9] J. Rasmussen, K. G. Larsen, and K. Subramani. Resource-optimal scheduling using priced timed automata. In *Proc. 10th Int. Conf. of Tools and Algorithms for the Construction and Analysis of Systems (TACAS'2004)*, volume 2988 of *Lecture Notes in Computer Science*, pages 220–235. Springer-Verlag, 2004. Available from World Wide Web: `http://www.springerlink.com/app/home/contribution.asp?wasp=9a0qbvyyrjdjt6rvmewp&referrer=parent&back`.