Exercise: Using SAT based arithmetic model checking for task deployment and scheduling problems

2007-05-24 Martin Fränzle, Andreas Eggers

---

1 Overview

In this exercise you will generate predicative encodings of scheduling and deployment problems for HySAT[1], a DPLL-based arithmetic constraint solver.

We first present a hardware architecture and task descriptions. Thereafter you will find a short introduction to HySAT. Your task will then be to formalize the relevant entities for a scheduling problem and to use HySAT in order to find out whether the system is schedulable. After adding another task to the system you use the tool to find out if the requirements for the hardware need to be changed in order to meet all deadlines of the extended system. A last step is to add a metric for the energy consumption of the system. The model checker is then used to find a schedule with minimal energy consumption with respect to that metric.

2 System description

2.1 Hardware

The system on which our tasks are intended to run has two types of memory:

- "normal" memory (MEM) of sufficiently large size and
- a very little amount of scratch pad memory (SPMEM).

SPMEM has the advantage that access to it is very quick and energy consumption for access to variables in SPMEM is also lower than for accessing variables in MEM. It is used instead of caches in order to allow easier prediction of the access time that must be taken into account when reading from and writing to memory.

Only one single-threaded CPU is available, i.e. only one task can be active at a time. There are no resources that can be locked by a task.

2.2 Tasks

Tasks are called periodically and do not depend on the results of each other. They have deadlines which may not exceed their periods. The worst case execution time C_i of a task consists of some base time BC_i and the time that it needs to access its variables. These variables can be assigned to either a fixed position in MEM or a fixed position in SPMEM. Swapping of variables from SPMEM to MEM in order to use SPMEM differently for different tasks is not possible.

2.3 Scheduling

The system is equipped with a fixed priority scheduler with preemption. Each task is thus given a priority at design time. Whenever a task with a higher priority than the task that is just running (if any) becomes activated, this task preempts the running task, which is suspended until the processor is free again.

In order to check whether this method yields a schedule in which no deadline is violated, one can calculate the response time of each task and check whether it is below the task's deadline D_i.

[1] `http://hysat.informatik.uni-oldenburg.de`

Let P_i denote the period of task i, C_i the WCET of task i, hp_i the set of tasks with higher priority than task i then

$$R_i = C_i + \sum_{k \in hp_i} \left\lceil \frac{R_i}{P_k} \right\rceil \cdot C_k$$

yields the response time of task i. This recursive formula is normally solved by searching a fixed point beginning with $R_{i,0} = C_i$ as initial candidate for R_i and then inserting $R_{i,0}$ into the recursive formula as R_i. If the resulting $R_{i,1}$ equals $R_{i,0}$ the fixed point is already reached, otherwise the calculation is continued with $R_{i,1}$ as a candidate. The search can be stopped when either a fixed point is reached or the deadline exceeded.

3 HySAT

The scheduling and deployment problems that are to be solved have to be written as predicative descriptions. The idea is to write down one large formula which is satisfiable if and only if a schedule exists that does not violate any additional constraints such as e.g. a maximum energy consumption. This large formula is then given to HySAT, which then searches for such a solution. As HySAT is designed for a very general class of non-linear systems, it cannot guarantee that a valuation is verified. You will therefore have to perform a manual check of the resulting valuation. In case the constraint system is unsatisfiable, this result can be guaranteed (modulo programming errors).

The HySAT input language and the operation of HySAT are described in the manual to which you will be given access. Please try out some of the examples to become familiar with the syntax and the way the results are presented.

4 Instances

The concrete values given here are not intended to be realistic. They have primarily been chosen in order to simplify manual calculations to check the plausibility of the results.

System parameters

Hardware	
MEM size	*sufficiently large, i.e. don't care*
SPMEM size	4 cells
MEM access time	4 cycles
SPMEM access time	1 cycle
Task 1	
number of accesses to T1 variable 1	10
number of accesses to T1 variable 2	3
number of accesses to T1 variable 3	2
number of accesses to T1 variable 4	6
period P_1	1200
basic WCET BC_1	140
deadline D_1	1000
Task 2	
number of accesses to T2 variable 1	5
number of accesses to T2 variable 2	40
number of accesses to T2 variable 3	1
period P_2	200
basic WCET BC_2	10
deadline D_2	100

Assignment 1:

Model the system as described so far in HySAT. Either assign fixed priorities to the tasks, e.g. using the deadline monotonic approach of giving the highest priority to the task with the most urgent deadline, or allow these priorities to be chosen by HySAT. Add constraints such that a solution of the constraint system gives a certificate for the existence of a schedule (modulo the uncertainty that HySAT's result could contain spurious valuations). Play around with some of the parameters, e.g. the SPMEM size or the access times. Also try to add additional conditions e.g. that a certain variable must not be allocated to SPMEM or that the response time of a task may not exceed a certain value (which is actually a change of the deadline).

Assignment 2:

Extend the model with the following task. Keep in mind that this may change the assigned task priorities and has an influence on the response time analysis.

Task 3	
number of accesses to T3 variable 1	5
number of accesses to T3 variable 2	7
number of accesses to T3 variable 3	4
number of accesses to T3 variable 4	6
number of accesses to T3 variable 5	24
number of accesses to T3 variable 6	100
period P_3	400
basic WCET BC_3	20
deadline D_3	300

Try out to find a schedule for this new task system. If no schedule can be found, find the minimum SPMEM size such that the system becomes schedulable (not caring about the fact that this would in practice probably mean to increase it by a power of 2).

Assignment 3:

Add a measurement for the energy consumption to the system you generated in the previous assignment. Each access to a variable in MEM costs 30 energy units, each access to a variable in SPMEM costs 2 energy units. If e.g. T1's variable 1 was stored in SPMEM and all other variables in MEM then each invocation of task 1 would cost $e_1 = 10 \cdot 2 + 3 \cdot 30 + 2 \cdot 30 + 6 \cdot 30$ (ignoring the base cost for the parts that are executed independently from variable accesses). As the tasks are invoked with different rates, it makes sense to generate a more abstract energy metric for the system:

$$E = \sum_i \frac{1}{P_i} \cdot e_i$$

with e_i being the energy consumption of task i. Add the necessary constraint to the model and then try to find a minimal value for E.