Deliverable CS5.2:
Report on VHS Case Study 5:
Sidmar Steel Plant at Ghent, Belgium

Introduction

Case Study 5 was included in the project as an example of an industrial plant where a realistic scheduling and resource allocation problem has to be solved. The plant model and the specifications to be met in order to ensure acceptable plant operation include both purely logical and hard real-time constraints. The quantitative performance measure, to be minimized, involves the time duration of the activities.

The size of the plant is not execssively large, but the number of states is nevertheless such that enumeration is excluded as a method for verification and for control synthesis. Therefore it provides the partners in the project with a realistic test case for evaluating the performability of various modelling approaches and of different verification and control synthesis tools for timed discrete event systems and for hybrid systems. Moreover the goal of scheduling is the minimization of a cost function - the time needed to produce a fixed sequence of batches of steel. As such this case study provides an example where verification of safety constraints must be combined with optimization of quantitative performance measures. The fact that the controller is implemented in a distributed way at SIDMAR ensures that a modular modelling approach used by the various academic research groups agrees well with the industrial implementation.

During the first year of the project several partners developed models for this steel plant: timed automata (within the UPPAAL framework, developed by BRICS and KUN), constraint logic programming (SIDMAR) and timed Petri nets (Ghent). Verification of the feasability of schedules was achieved for models of moderate size (the number of batches in a sequence of tasks to be completed being the major limiting factor).

During the second year of the activity in case study 5, several extensions of this model have been investigated, and verification of schedules for larger plant models has been achieved. More importantly automatic schedule generators with guidance by the users' intuition have been realised based on an UPPAAL model and based on a constraint logic programming model. The schedule developed by Sidmar using constraint logic programming has been successfully implemented in practice. The BRICS Group has been able to implement the schedule on a fairly realistic hardware simulator of the steel

plant.

It has been observed that computation time depends heavily on the order in which different branches of the tree of reachable states is searched, and that sometimes a lot of backtracking causes excessive computational costs. In order to speed up the search for an optimal schedule (in particular in order to make on-line generation of a schedule practically feasible) it is necessary to use recursive algorithms for searching in an efficient way through the space of all schedules. Several proposals for using iterative optimization strategies have been made by the groups in Dortmund and Ghent. The group in Dortmund proposes the combination of a constraint programming model with heuristic techniques (such as branch and bound) for finding a small set of good schedules. The group in Ghent proposed a dynamic programming-like synthesis of an optimal schedule based on a modular plant model, where each module is a timed Petri net with input and output transitions.

The results of the work on case study 5 have shown that verification is feaible for a fairly large plant, and that this can be very useful in automatically synthesizing good solutions for scheduling and resource allocation problems in industry. The work has shown a nice convergence between the work in industry and in acedemia. Moreover the work has pointed the way to a number of interesting open questions on verification and on control synthesis which will be further investigated in other workpackages during year 3 of the project.

Progress

Scheduling and resource allocation paradigm

The steel plant operation studied in CS5 provides the VHS partners with a useful paradigm of a scheduling and resource allocation problem. Starting from N batches of raw molten pig iron available as input to the convertor, the steel plant produces N batches of molten steel, each batch having its specified quality defined by its recipe. A recipe specifies the quality of the steel by specifying the different resources being used consecutively, and the time duration of the use of each resource in the plant. The order in which the N batches of different qualities of steel arrive at the continuous casting machine is prespecified.

The cost of the plant operation is the total time necessary for executing the complete sequence of batches. In other words the performance measure to be minimized is the execution time of the start of the casting of the last batch in the prespecified sequence. The execution times of tasks can

be specified via upper and lower bounds, or via probability distributions. In the first case the cost to be minimized is the worst case performance measure. In the second case the cost to be minimized is the average of the performance measure.

The plant model has to describe the prespecified sequence of qualities arriving at the continuous casting machine, the sequence of resource requirements of the different recipes, the limited resource availability in the plant, and constraints on the maximum life time of each batch. The various recipes have to be executed using a limited number of resources, with some tasks of the recipe requiring simultaneous and exclusive access to more than one resource. In order to achieve this, starting times of certain tasks must be scheduled appropriately. Some tasks can be executed on different resources, and hence resource allocation is required. Various precedence constraints, and some real time constraints have to be satisfied by a valid schedule.

The control synthesis problem to be solved consists in finding a schedule, which achieves the smallest value for the total production time of a sequence of batches (worst case or average) within the set of all valid schedules. The schedule selects the time when tasks are to be started, subject to plant model constraints, and allocates resources for carrying out this task whenever there is a choice between different resources.

The scheduling and resource allocation problem for the case study of the steel plant amounts to taking the following main decisions. A first scheduling decision concerns the times at which the conversion of a new load of pig iron is started in a convertor which is ready for a new load. After conversion has been completed, this raw molten steel is tapped from the convertor into a steel ladle, provided various constraints are met, for example an empty steel ladle is available and can be transported on a transfer car towards the metallurgical position where the first treatment of the raw molten steel will be carried out. A choice has to be made on which position this first treatment will be done. Once this first treatment is completed, this same ladle is used to transfer the batch of molten steel to another position where the second phase of the treatment is carried out. Again the position where this second phase of the treatment is to be carried out must be selected. Eventually the steel plant transforms the batch of molten steel contained in the ladle into steel of a specified quality by executing the various tasks of the recipe corresponding to the specified quality of steel on the different metallurgical treatment positions. After all the treatments prescribed by the recipe have been executed the batch of steel is ready for casting at the continuous casting machine. At that time the controller has to select the appropriate casting speed (ensuring that the casting time is between its

minimum and its maximum specifications, and ensuring that the next batch can arrive in time for uninterrupted, continuous casting). In order to enable all these transitions it is also necessary to schedule movement of both cranes available in the plant.

The model presentation originally emphasized

- the availability of the resources for steel making (steel ladles for tapping the convertor, metallurgical treatment positions for carrying out various metallurgical treatments, buffering positions, continuous casting machine),
- the availability of the transport resources (overhead cranes for moving steel ladles from one track to another track, transfer car for moving steel ladles along a track),
- and the recipes according to which different qualities of steel have to be made (expressing the order in which various resources are needed for producing the quality corresponding to the recipe; each recipe also specifies the real time properties of the plant by expressing for each required treatment for how long each resource is needed).

These requirements specify the set of possible sequences of events that are allowed to occur in the plant. The plant operation is also characterized by the requirement that the order of casting different qualities of steel agrees with a predetermined ordering, which should be adhered to strictly for achieving acceptable operations. This was emphasized as a logical constraint on the plant operation. On the other hand it was emphasized that there is also a real time specification requiring that each batch of steel must be produced within an interval of time that is limited by an upper bound. This upper bound must insure that the batch of steel has not cooled down too much when casting starts. Finally the continuous casting requirement also imposes a real time constraint. In order to verify a schedule it is of course necessary to explicitly model the rules for scheduling and for resource allocation, as they are used by the plant operator.

Schedule generators based on verification tools

A partial specification of a schedule has been included in the timed automaton model developed by Fehnker (Nijmegen) [Feh99a],[Feh99b]. It has been possible to verify existence of a feasible schedule by using this model as an input to the verification tool UPPAAL. A complete schedule that fixes all the decisions to be taken in such a way that it satisfies all the specifications,

has been generated automatically by the same tool. This schedule is also reported in the same reference. In [HLP99] it is shown that schedules can be generated for larger, more realistic plant models and for larger values of N by combining the automatic reasoning of UPPAAL with the users' intuition.

This work was further extended in the second year by the BRICS Group [HLP00]. Schedules can be generated for larger plant models and for larger values of N by combining the UPPAAL model verification tool and some heuristics which automatically eliminate schedules which are known on intuitive grounds to be inefficient (such as schedules with a lot of unnecessary crane movement, or schedules with idle resources when it is obviously possible to move forward certain task execution without compromising system constraints). The quality of the schedules thus obtained have been tested extensively on a hardware model of the steel plant, built on the basis of the LEGO Mindstorm system [IKL$^+$00]. In the joint work [BHV00] it has been shown that verification of the validity (and hence also generation) of schedules can be made a lot faster by proper choice of the order of selecting different branches in the set of reachable states. In particular it has been found that by using parallel search the speed of the verification tool increases by many orders of magnitude, thanks to the fact that a better selection of consecutive branches is feasible.

Within the Systems and Models Group of Sidmar a constraint logic programming model has been developed and implemented for obtaining good schedules of the tasks in the steel plant [Sid99]. All the constraints of the resources of the plant (except for the constraints on the crane availability - these crane constraints are handled via a posteririori simulation) are encoded in the constraint sets, used as input for an ILOG programme. A heuristic algorithm for searching valid schedules is then used, and a simulation routine is implemented

Recursive algorithms for generating optimal schedules

The time required for finding a good schedule depends very much on the order in which the successive schedules under consideration (all member of the set of all valid schedules) is searched. When a schedule under consideration is found to be infeasible, it is very important to decide how one can select the next schedule to be considered. In particular one has to decide how to backtrack through the large tree of constraints. This consideration led project members in Dortmund and Ghent to consider recursive optimization algorithms. These algorithms use some specific criteria in order to select the next schedule to be considered as a candidate. Several heuristic or theoreti-

cally based rules can be used in order to ensure that the next schedule under consideration is valid most of the time, and is such that it decreases the cost (moves in the direction of optimizing the performance measure).

In [Sto00] an approach based on Constraint Programming is proposed for the plant model. A nearly optimal schedule is then found in the following way. During the n-th iteration each constraint locally propagates information about the possible values for local variables, and this information is used to reduce the set of possible valid $(n+1)$-th schedules as much as possible. In order to reduce the cost as quickly as possible a branch-and-bound algorithm is used for the selection of this $(n+1)$-th schedule under consideration. This approach has shown good results for generating schedules up to length 30.

In [BM00], the model is described in the form of a graph of interconnecting modules, each module represented as a timed Petri net (or a timed automaton) with input and output transitions, which express the interaction between modules. It turns out that the major activities in the steel plant (conversion, metallurgical treatment, and casting) can be expressed via an acyclical graph of modules. This allows generation of good schedules moving backward through the acyclic graph of modules. Within each module an optimal schedule is obtained by forward recursion, taking interaction with successor modules (as defined in the graph of modules) into account. The validity of the schedule thus obtained at each iteration is verified for the other (crane related and empty ladle related) modules. Several rules are then suggested for finding the next schedule under consideration. In [Boe00] this approach is extended to more general plants where the graph of modules required for generation of schedules is cyclical.

Contributions to the case study

The work (both for year 1 and year 2) on case study CS5 has been reported in the following reports.

René Boel and Geert Stremersch(Ghent): VHS Case Study 5: modelling and verification of scheduling for steel plant at SIDMAR: [BS99b]

This report provides a verbal description of all the plant resources, of the recipes, and of the other specifications of the plant operation. It also provides a very brief outline on the propietary scheduling rules currently in use at the Sidmar steel plant. The description starts from the plant layout. The

report presents in detail the transportation system, the convertor operation, the metallurgical treatment stations, the continuous casting machine; next the report describes the recipes corresponding to the different qualities of steel to be produced, the sequence of recipes to be executed. The report also lists the various purely logical as well as the real-time specifications that should be met by the plant operation. It emphasizes the need to produce batches of steel in such a way that they arrive at the continuous casting machine according to a sequence of qualities that is specified in advance. This ordering in which the qualities must arrive at the continuous casting machine is specified in advance by a higher level optimization routine. This sequence is to be followed strictly by the schedule used for the plant operations. The size of the full model is such that the number of states of the plant is very high (much more than 10^{20}) so that enumeration is not a feasible approach for verification and synthesis of schedules. There are approximately 30 different qualities of steel but these can be produced by minor variations on about 6 different recipes. The sequence of steel qualities as specified by the higher level plant optimization has a typical length of about 60 batches. Each load passes through approximately 10 to 20 different stages (depending on the recipe under consideration) between tapping at the convertor, and casting the molten steel at the continuous casting machine.

Ansgar Fehnker: Scheduling a Steel Plant with Timed Automata(Nijmegen): [Feh99a]

. This report uses Timed Automata (TA) to model the steel plant at Sidmar, Ghent, Belgium. The timed automaton formalism can be used to model real-time requirements of systems in a natural way. In recent years several tools, such as UPPAAL and KRONOS, have become available for automatic model checking based on timed automata. This case study confirms that these tools can be used for realistic applications. The steel plant is modelled as a network of timed automata, ready to serve as input for the UPPAAL tool. The different components of the model are combined using binary synchronization, and using committed locations. The model distinguishes components describing batches to be produced according to a given recipe and a component for the plant itself. A test automaton is introduced to represent the specified sequence of qualities of steel to be produced. This test automaton is just a linear automaton proceeding from the first quality of steel to be produced to the last quality to be produced. A feasible schedule for the plant operation exists provided the final state of the test automaton is reachable from the present state of the plant model. The report treats a few

examples of small models that have been analyzed. For a simple example with a sequence of only 3 batches, and with the use of only one crane, the report lists a complete schedule, that has been generated automatically by the tool. Larger examples with up to 7 batches in the sequence, and with both cranes active, have also been analyzed. These results are quite encouraging. They suggest that it will be possible to achieve the final goals of case study 5 by carrying out further work during the second year of the project.

A. Fehnker: Scheduling a Steel Plant with Timed Automata. [Feh99b].

Scheduling in an environment with constraints of many different types is known to be a hard problem. We tackle this problem for a integrated steel plant in Ghent, Belgium, using Uppaal, a model checker for networks of timed automata. We show how to translate schedulability to reachability, enabling us to use Uppaal's model checking algorithms.

T. Hune, K. Larsen, and P. Petterson:Guided Synthesis of Control Programs using UPPAAL for VHS CS5:[HLP99]

This reports presents a more realistic timed automaton model of the steel plant at Sidmar by including the time delays of the overhead crane transportation system, which were ignored in [Feh99a]. The refined model should allow the extraction of a more detailed control program for the actual plant. The refined model makes the straightforward application of UPPAAL as a tool for extracting a valid control program more difficult. To deal with this unavoidable problem the authors introduce a method, allowing the user to *guide* the synthesis according to certain chosen strategies. Each of these strategies will contribute to a reduction of the search space, but in contrast to fully automatic reduction methods it is up to the users' intuition to *guarantee* preservation of schedulability. Examples of guiding strategies in the steel plant are: removal of useless back-and-forth movement of loads; limiting the choice of paths between distant points; limiting the maximum amount of time for some tasks. This guidance is achieved by using guards on transitions, after augmenting the model with integer variables. The report provides a realistic schedule obtained in this way. A hardware simulator of the steel plant has also been built in order to allow experiments with various(good as well as bad) schedules.

R. Boel and G. Stremersch (Ghent): Timed Petri Net Model of Steel Plant at Sidmar(Ghent): [BS99c]

This report presents a model using the semantics of timed Petri nets [BS99a] for representing the plant constraints and some of the specifications of the model. The plant model is represented via the synchronous execution of two timed Petri net components. One component represents the overhead cranes. The other component represents the resource requirements of the various recipes, and the availability of transfer cars for steel ladles. Different recipes are represented by different parallel branches in this timed Petri net component. Common transitions can only be executed when they are enabled in both components. The time requirements for the different treatments are expressed in this model by stating that a transition must be executed within a given interval after it has become state enabled. The specification that batches of steel arrive at the continuous casting machine in a specified order of qualities is expressed in this model via a set of linear inequalities involving execution times of transitions in the recipe model. The requirement that each batch gets completed within a certain time delay is expressed in this model by upper bounds on the difference between the execution times of different transitions. Other constraints requiring that cranes do not collide, that each treatment station can contain only one steel ladle at a time, etc. are expressed by inequalities on the marking of the Petri net.

Thomas Hune, Kim Larsen, and Paul Pettersson (BRICS): Guided Synthesis of Control Programs Using UPPAAL: [HLP00]

This paper, and its accompanying web page, give some additional information about our work on synthesizing control programs for a LEGO® plant. The plant is based on parts of an existing steel production plant, SIDMAR in Belgium and is a case study of the VHS project.

The first step of generating a control program for the plant is to find a schedule for the plant. To find a schedule for the plant we have modeled the plant using timed automata. The scheduling problem can be stated as a reachability question which can be analyzed by the tool Uppaal. The Uppaal model of the plant consists of a number of automata:

The automaton representing the behavior of a batch.

A recipe using machines of type one and two.

A recipe using machines of all type.

The automaton representing the 'upper' crane.

The automaton representing the 'lower' crane.

The automaton representing the casting machine.
The test automaton.
A 'controller' only used to make synchronization visible in a trace.

Torsten Iversen et al. (BRICS): Model-checking Real-time Control programs – Verifying LEGO Mindstorms Systems Using Uppaal, [IKL+00]

In this paper we present a method for automatic verification of real-time control programs running on $LEGO@RCX^{TM}$ bricks using the verification tool UPPAAL. The control program, consisting of a number of tasks running concurrently, are automatically translated into the timed automata model of UPPAAL. The fixed scheduling algorithm used by the $LEGO@RCX^{TM}$ processor is modeled in UPPAAL, and supply of similar (sufficient) timed automata models for the environment allows analysis of the overall system using the tools of UPPAAL. To illustrate our techniques we have constructed, modeled and verified a machine for sorting LEGO@ bricks by color.

Mario Stobbe (Dortmund): Results on Scheduling the SIDMAR Steel Plant Using Constraint Programming, [Sto00]

This report is a result of the participation of Process Control Laboratory of the University of Dortmund in the research project "Verification of Hybrid Systems" (VHS). It deals with the application of Constraint Programming for synthesising near optimal schedules for the steel plant presented in the case study 5 of the VHS project. The report is organised as follows. First we briefly describe the main constraints of problem, the objective of the scheduling task and the basic assumptions we have made. In the second chapter, we briefly review the applicability of two well known models for scheduling plants in the process industry. We mainly focus on the model size and the description of the recipes. Thereafter we propose a formulation using Finite Set Constraints leading to a model of limited size. In chapter three, the solution approach is presented and some of the obtained results are given and discussed.

René Boel and Francisco Montoya (Ghent): Modular synthesis of efficient schedules in a timed discrete event plant, [BM00]

This paper treats optimal scheduling in large timed discrete event systems as a supervisory control problem. This paper considers a case study where

a schedule must be synthesized which completes a sequence of tasks in the shortest possible time, while satisfying all the constraints in the model, and satisfying some hard outside constraints. The paper emphasizes the importance of describing the plant via a graph of interacting modules, Each module can be represented using a different mathematical formalisms. An algorithm is proposed for efficiently finding an optimal schedule by first finding an optimal schedule in each module separately, for the largest subset of modules which form an acyclic graph. A model of a steel plant is used as an example.

René Boel (Ghent): Automatic Synthesis of Schedules in a timed discrete event plant [Boe00]

This extended abstract discusses some ideas concerning the modular control synthesis of optimal schedules for large plants. In particular it is emphasized that there is usually a distinction between foreground events and background events. We compare two cases, one for a steel plant, the other an urban traffic network. The foreground events include all the events to be scheduled. In the steel plant this corresponds to the start of different operations in the steel making. In the urban traffic this corresponds to the red/green switches of traffic lights at the intersections along the major route(s) carrying foreground traffic. The execution time of all the schedulable events determines the state of the foreground plant and this also determines the cost of the plant operation. This foreground plant model usually consists of several interacting components or modules. Timed Petri nets are a good tool for describing such input-output timed discrete event modules. It is shown that the complexity of the scheduling problem is related to the structure of the graph describing the interconnection between the foreground modules. Scheduling is relatively easy when this foreground graph is acyclic, because backward recursion can be used. When the foreground graph contains cycles then iterative techniques are necessary for scheduling. The evolution of the background plant modules, decribing e.g. crane movements in a steel plant or interferring traffic from side streets in an urban traffic network, must also be desribed using several modules. This abstract argues that these background modules can often be represented via stochastic automata (e.g. stochastic Petri nets). In any case the interaction between different modules can be described via sensitivity analysis. This abstract refers to earlier work by the author where it is shown that sensitivity of a cost function with respect to changes in execution time of input events at a module can be calculated using one single simulation run.

Sidmar: Evaluation report on ILOG Scheduler[Sid99]

This internal report of Sidmar describes the use of Constraint Programming and of ILOG Solver as tools for generating feasible schedules for a steel plant. Strict separation of model and solver, and flexibility allowing quick modification of the plant are seen as major advantages. Problems sometimes occur when the solution requires a lot of backtracking in its search through the space of all constraints.

Conclusions

Case Study 5 has provided the project partners with an industrial model which proved amenable to analysis by the tools under consideration. Several modelling formalisms have been shown useful in verifying and in synthesizing schedules for the steel plant. The good cooperation with the industrial partner has allowed thorough analysis of this case. The model has turned out to be a paradigm for many interesting scheduling and resource allocation problems. It will serve as a basis for further work in the third year of the project within the workpackages of process analysis and modelling formalisms.

An important aspect of the implementation of good schedules where the present model based approaches have not yet been used to their full advantage is the problem of building a "safety net" against all the possible errors in the on-line observed data. This is an important issue for the on-line implementation of any scheduler. Fault detection based on formal models could therefore be an important topic for consideration during the final year of the project.

The main results can be summarized briefly as follows.

1. Timed Automata, timed Petri nets and constraint programming, and the corresponding tools have been applied successfully to the analysis and schedule synthesis for the problem of case study 5.

2. The academic research groups in the project have been able to recover very quickly scheduling algorithms similar to those used by Sidmar, on the basis of long years of experience, thanks to the use of formal modelling techniques, such as timed automata.

3. The use of formal models allows a clear separation between model building and schedule synthesis. It also allows quick synthesis of new schedules whenever there is a modification to the plant.

An important open question for industrial application of the results obtained in this case study is the following. Timed automata and timed Petri nets are quite abstract models, and therefore perhaps not very suitable for immediate introduction into industrial applications. Developing automatic translation tools between timed automata (or timed Petri net models) and models more easily used in industry (such as constraint programming) might therefore be very useful.

Bibliography

[BHV00] Gerd Behrmann, Thomas Hune, and Frits Vaandrager. Distributed timed model checking - How the search order matters. In *Proc. of 12th International Conference on Computer Aided Verification*, Lecture Notes in Computer Science, Chicago, Juli 2000. Springer-Verlag.

[BM00] R.K. Boel and F. J. Montoya. Modular synthesis of efficient schedules in a timed discrete event plant. In *submitted for CDC2000*, 2000.

[Boe00] R.K. Boel. Automatic synthesis of schedules in a timed discrete event plant. In *Proceedings of ADPM2000*, 2000.

[BS99a] René Boel and Geert Stremersch. Semantics of timed Petri nets. draft, 1999.

[BS99b] René Boel and Geert Stremersch. VHS case study 5: modelling and verification of scheduling for steel plant at SIDMAR. draft, 1999.

[BS99c] René Boel and Geert Stremersch. VHS case study 5: Timed Petri net model of steel plant at SIDMAR. draft, 1999.

[Feh99a] Ansgar Fehnker. Scheduling a steel plant with timed automata. Technical Report CSI-R9910, Computing Science Institute Nijmegen, 1999.

[Feh99b] Ansgar Fehnker. Scheduling a steel plant with timed automata. In *Proceedings of the 6th International Conference on Real-Time Computing Systems and Applications (RTCSA99)*, pages 280–286. IEEE Computer Society, 1999.

[HLP99] Thomas Hune, Kim G. Larsen, and Paul Pettersson. Guided synthesis of control programs using UPPAAL for VHS case study 5. VHS deliverable in Workpackage CS.1.1, 1999.

[HLP00] Thomas Hune, Kim G. Larsen, and Paul Pettersson. Guided synthesis of control programs using Uppaal. In Ten H. Lai, editor, *Proc. of the IEEE ICDCS International Workshop on Distributed Systems Verification and Validation*, pages E15–E22. IEEE Computer Society Press, April 2000.

[IKL+00] Torsten K. Iversen, Kåre J. Kristoffersen, Kim G. Larsen, Morten Laursen, Rune G. Madsen, Steffen K. Mortensen, Paul Pettersson, and Chris B. Thomasen. Model-checking real-time control programs — Verifying LEGO mindstorms systems using Uppaal. To appear in ECRTS'2000., 2000.

[Sid99] Sidmar. Evaluation report on ilog scheduler. internal report, Systems and Modellling Division, Sidmar., 1999.

[Sto00] M. Stobbe. Results on scheduling the sidmar steel plant using constraint programming. Internal report, 2000.