Acta Polytechnica Vol. 48 No. 3/2008

Visualization and Simulation in Scheduling

R. Čapek

This paper deals with the representation of scheduling results and it introduces a new tool for visualization and simulation in time scheduling called VISIS. The purpose of this tool is to provide an environment for visualization, e.g. in production line scheduling. The simulation also proposes a way to simulate the influence of a schedule on a user defined system, e.g. for designing filters in digital signal processing. VISIS arises from representing scheduling results using the well-known Gantt chart. The application is implemented in the Matlab programming environment using Simulink and the Virtual Reality toolbox.

Keywords: Scheduling, visualization, simulation, Matlab, VISIS.

1 Introduction

Scheduling theory plays an important role in optimization of resources, and is used in many manufacturing and service industries. In the last fifty years, many optimal and heuristic algorithms have been proposed, but there is growing demand for transparent and realistic representation of results in scheduling. The objective of visualization and simulation is to make these theoretical results accessible to non-experts in scheduling theory. Especially production scheduling and planning needs to be represented in a transparent form. The goal of this work is therefore to extend the existing TORSCHE Scheduling Toolbox for Matlab [1] by a new tool for graphic visualization of time schedules.

Scheduling optimizes the utilization of resources with given constraints in time. In other words, scheduling solves the problem how to assign given resources to given tasks in time [2]. Production scheduling is a branch of scheduling mostly aimed at automated production lines and industrial production in general [3]. In this paper, visualization means graphic representation of a schedule in time, whereas simulation monitores the influence of some parameters on a whole system.

This work emerges from the TORSCHE Scheduling Toolbox for Matlab [http://rtime.felk.cvut.cz/scheduling-toolbox/], which provides data structures and algorithms for time scheduling. Therefore, the whole project is realized in the Matlab programming environment [http://www.mathworks.com/]. One of the related tools is TrueTime [4] – a Matlab based tool for real-time simulation for wide spectrum of problems, e.g. digital filters, embedded systems and wireless networks. For visualization, OpenGL (Open Graphics Library) [http://www.opengl.org/] is a standard specification defining a cross-platform API for writing applications that produce 2D and 3D computer graphics. This means that visualization can be realized by OpenGL at any operation system. On the other hand, Matlab includes the Virtual Reality toolbox, which is also a sufficient tool for visualization of scheduling results. From the scheduling area there are also some closely related works. Optimization using simulation was described by Fishman [5], and the use of simulation based optimization in real production was briefly described by Manlig and Sramek [6]. Visualization in scheduling has been studied at Karlsruhe University [7] and an application for visualization of process scheduling has been developed there.

The main goal of this paper is to present the application for visualization and simulation of scheduling results in the Matlab environment – VISIS (VIsualization and SImulation in Scheduling). This application uses the Matlab-based simulation environment Simulink and the Virtual Reality toolbox for graphic visualization. Two areas of usage are considered: simulation for monitoring the influence of scheduling on the system function (e.g. for digital filters), and time visualization (e.g. graphic represPentation of execution on a production line in time). This tool will be freely available in the next version of TORSCHE. To the best of our knowledge, there is no such a tool providing visualization of scheduled processes in this range.

This paper is organized as follows: Section 2 provides the basic notation used in scheduling theory. Section 3 describes the implementation of VISIS. In Section 4, examples of simulation and visualization are provided and a comparison with TrueTime is shown. The last section concludes the work.

2 Representation of results in scheduling

Scheduling problems can be divided into three categories: resources (processors or machines), constraints and criterions. Generally accepted notation of the problem has the form $\alpha|\beta|\gamma$, where α stands for types of processors used, β represents tasks and characteristics of resources, and γ denotes the optimality criterion [2]. For example, $1|r_j, \sim d_j|C_{\max}$ represents the problem with one processor (resource), given release date and deadline for each task, while the objective is to minimize the latest completion time. The $C_{\max}$ value is the most frequently used criterion, because it represents the throughput of the system [8]. Generally, scheduling is NP-hard problem. Polynomial algorithms are therefore known only for the limited number of problems. This leads to an exponential rise in the time needed to find the optimal solution in dependence on the number of input tasks.

The most common graphic representation of scheduling results is the Gantt chart [http://www.ganttchart.com/], first established by Henry Gantt in 1910. The Gantt chart has discrete time values on the x axis and processors on the y axis (see Fig. 1). Tasks are represented by a rectangle area on the intersection of the appropriate processor and the assigned time. The form of the Gantt chart is identical for all time

schedules represented by the start times and processing times of the tasks. For the hoist scheduling problem [9], there is another way to display the results (see Fig. 2). It is a chart with discrete time values on the *x* axis again and on the *y* axis there are tanks where the material is processed. The time schedule is then represented by lines denoting the moves of the hoists, one type for loaded hoists, one for empty hoists, and one for material storage in tanks. This type of chart is special for the hoist scheduling problem, and it gives a better idea of the final result, although understanding is at first quite difficult. Visualization should arise from the individual problem definition instead of one general form.

3 Implementation

As mentioned above, this project is realized completely in the Matlab environment, and the output of the application is a Simulink scheme. Graphical objects for visualization are created in VRedit (part of the Virtual Reality toolbox for Matlab) and the final visualization is displayed using the Virtual Reality toolbox. The VRedit environment allows us to define basic geometrical objects, text, background, textures and complex predefined objects. Each object in Virtual Reality has its own set of variable parameters. The numerical value of any parameter can be directly changed from Simulink.

The VISIS implementation provides several functions available for users. For maximum simplicity of usage, the Simulink scheme is generated automatically. This output scheme contains one masked subsystem representing the control system. In the case of visualization, there is another block referencing the predefined virtual reality world. The mask of the control subsystem has inputs and outputs with user-defined names and sizes. The core of the control subsystem is the S-function block, which contains the main control function. This function updates the outputs according to the given schedule and the actual values of the inputs. This control function is also generated automatically, and all needed external data is created in the Matlab workspace before the simulation begins. The S-function block has only one input and output port as default so the in/out signals are integrated/divided to reach user defined number of inputs and outputs. This subsystem is then masked as one block with appropriate ports. The Simulink scheme and code for the S-function block are both generated as text files from the prepared templates. The control function is called for each sample of Simulink, and the outputs are updated according to the schedule and the actual Simulink time.

All implemented functions can be called as standard Matlab functions. Users of VISIS are expected to have basic knowledge of using the TORSCHE toolbox, and to be able to create their own project in the Virtual Reality toolbox. The first step in simulating or visualizing is to create a set of tasks (function taskset). Then the code of the operations has to be assigned to the tasks by the *adduserparam* function. This function reads data from the given text file, and the user-defined code is assigned to appropriate tasks in the taskset according to the format of the text file. The next step is to define the input and output ports for the control block and inputs of the Virtual Reality block, if needed. Then the set of tasks has to be scheduled using an appropriate scheduling algorithm and all created structures are passed to the main function *taskset2simulink*, which creates the Simulink scheme and the main control function. Any other block can be added to the scheme before simulation begins. The application checks the control function after it has been created, and if there is any structural error, a warning is displayed. Then a visualization can be seenin the Virtual Reality world, with the standard possibility to save any frame or video stream during the simulation.

4 Examples and experimental results

4.1 Hoist scheduling visualization

The hoist scheduling problem is chosen as an example for visualization. The classic representation of one period by the Gantt chart is shown in Fig. 1, where each task represents one move of a loaded hoist.

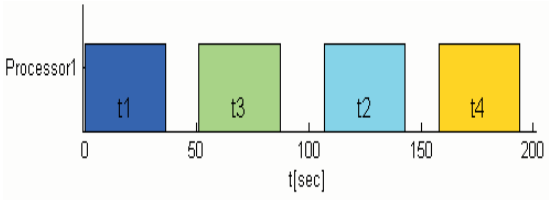

Fig. 1: Gantt chart for hoist scheduling

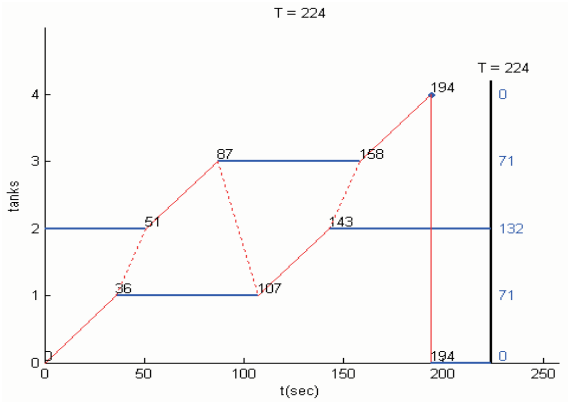

Fig. 2: Special chart for hoist scheduling

This representation does not reflect empty hoist moves. Temporary stays of the material in the tanks are also not clearly visible.A schedule of material moves is displayed in Fig. 2. This representation gives a better idea about the realization of the schedule, and it also displays a sequence of

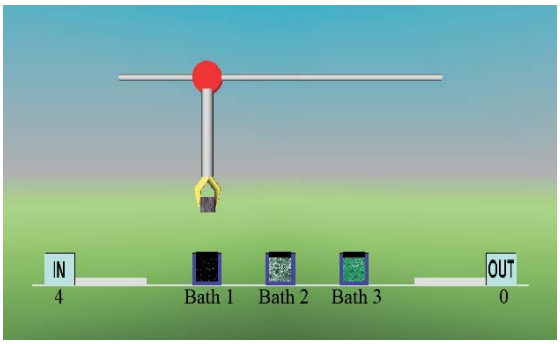

Fig. 3: Frame of visualization

several periods. Fig. 3 shows one frame of visualization by VISIS.

4.2 Digital filter simulation

Simulation of the DSVF-Digital State Variable Filter [10] is taken as an example of the simulation capabilities. This filter is formed by a set of arithmetic equations, which are repeated in a never-ending loop. Each elementary arithmetic operation has to be assigned to one task in accordance with the requirements of the scheduling algorithm requirements. Then the problem with the precedence constraints has to be scheduled and the resulting schedule can be passed to the *taskset2simulink* function. After the Simulink scheme has been generated, an appropriate signal generator and some display unit can be added. Then simulation is ready to start. The corresponding scheme is shown in Fig. 4, and the input and output signals of the modeled filter are displayed in Fig. 5.

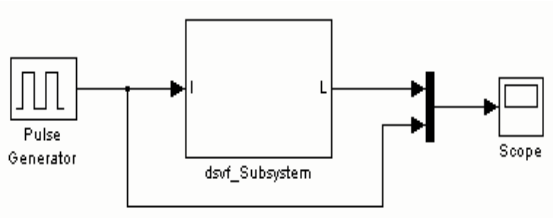

Fig. 4: Output Simulink scheme

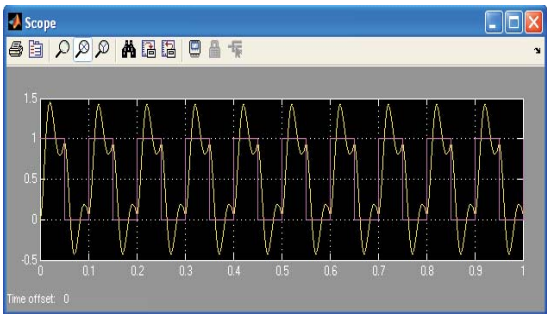

Fig. 5: Input and output signal

4.3 Experimental results

As mentioned above, generation of both the Simulink scheme and the control function code is text-based, so the time complexity of generating in dependence on the number of input tasks or on code length is approximately linear. Simulation by VISIS needs approximately 80 % of the time needed for the same example realized using the TrueTime library. In addition, the time needed for one second of simulation with 220000 samples per second is approximately 32 seconds in TrueTime and 26 seconds in VISIS.

5 Conclusions

This work has two areas of use: in discrete simulation (e.g. in digital signal processing) and in visualization of scheduled problems. VISIS is planed as an extension of a future version of the TORSCHE Scheduling Toolbox for Matlab, which can be freely downloaded. The application can be used for presentations, for educational purposes or as an optimization tool when clear representation of the results is needed as a feedback for scheduling.

The simulation of the digital filter is faster than in the TrueTime library, since VISIS is optimized for simulations of time schedules. The main advantage of VISIS is easier problem definition and simple usage.

Acknowledgments

This work was supported by the Ministry of Education of the Czech Republic under Research Programme MSM6840770038.

References

[1] Šůcha, P., Kutil, M., Sojka, M., Hanzálek, Z.: TORSCHE Scheduling Toolbox for Matlab. In *IEEE International Symposium on Computer-Aided Control Systems Design*. Munich, Germany, 2006.

[2] Blazewicz, J. et al.: *Scheduling in Computer and Manufacturing Systems*. Springer, 1993.

[3] Herrmann, J. W.: *Handbook of Production Scheduling*. Springer, 2006.

[4] Ohlin, M., Henriksson, D., Cervin, A.: *TRUETIME 1.4 – – Reference Manual*. Department of Automatic Control, Lund University. 2006.

[5] Fishman, G. S.: *Discrete-Event Simulation: Modeling, Programming, and Analysis*. Springer, 2001.

[6] Manlig, F., Šrámek, M.: Řízení výrobních zakázek s podporou počítačové simulace. In: *Průmyslové inženýrství*, 2003.

[7] Wittstein, H., Zoller, H., Lieflander, G.: *Visualization of Process Scheduling*. Universität Karlsruhe, Department of Computer Science. http://i30www.ira.uka.de/teaching/course documents/processscheduling/

[8] Crama, Y., Kats, V., van de Klundert, J., Levner, E.: Cyclic Scheduling in Robotic Flowshops. In: *Annals of Operations Research*, 2004.

[9] Manier, M. A., Bloch, Ch.: A Classification for Hoist Scheduling Problems. In: *International Journal of Flexible Manufacturing systems*, 2003.

[10] Matějíček, D.: *Optimalizace algoritmů pro FPGA*. Diploma thesis, CTU Prague, 2007.

Ing. Roman Čapek
phone: +420 776 716 588
e-mail: capekr1@fel.cvut.cz

Department of Control Engineering

Czech Technical University in Prague
Faculty of Electrical Engineering
Technická 2
166 27 Praha 6, Czech Republic