

Scheduling Toolbox for Use with Matlab

M. Kutil

kutilm@fel.cvut.cz

Department of Control Engineering, Faculty of Electrical Engineering,
Czech Technical University, Technická 2, 166 27 Prague 6, Czech Republic

TORSCH (Time Optimisation of Resources, SCHEDuling) [4] is a Matlab scheduling toolbox which proposes a background utilities for easy develop (rapid prototyping) and implementation of scheduling algorithms. Basic algorithms which are frequently used or which can be implemented as a part of complex algorithms are prepared too. The background utilities are divided into two categories. First one provides routines for classical scheduling problems implementation. Second one is intended for graph theory algorithms support. Both parts are closely related.

Classical scheduling problems utilities are described in [1]. Briefly, we can mention that main utilities are objects representing tasks and set of tasks. Task is a basic term in scheduling and describes any unit of work that is scheduled and executed by the system. The task is described by the following parameters: Name, Processing time, Release time, Deadline, Due date, Weight and Processor. Set of task is represented by taskset object which collects individual tasks and some additional parameters for example precedence constrains, user parameters, graphics representation parameters and so on. Newly, you can create taskset directly, without tasks preparing previously. This is possible by implication “dot-parentheses” syntax implementation to the object properties access. This access method is implemented for all objects in the scheduling toolbox and provides high comfort for users using Matlab console.

In new version some new scheduling algorithms are implemented and extended. Firstly, refer to the extended List scheduling algorithm. Basic idea of the algorithm is to sort tasks into list and next step by step take the task from list and assign them to processor. New modular system allows to extend the algorithm with external heuristics which order tasks in the list before releasing to the processor. Implemented heuristics are: earliest starting time (EST), earliest completion first (ECF), longest processing time (LPT) and shortest processing time (SPT). Verbose mode and unrelated processors support supplement List scheduling to be fully user friendly algorithm. Brucker’s algorithm is the next one new scheduling algorithm. This one demonstrates work with precedence constrains between tasks, especially precedence constrains in “in-tree” form [2]. Except traditional scheduling algorithms [2], TORSCH provides some modern special algorithms. One of them is an algorithm in literature called scheduling with positive and negative time-lags. This algorithm schedule tasks with release date and deadline related to the start time of another tasks. Cyclic scheduling [3] is next modern algorithm which is suitable for many activities e.g. in automated manufacturing or parallel computing. Cyclic scheduling means that tasks on machines are repeated cyclically. One repetition is usually called iteration and common objective is to find a schedule that maximizes throughput. Last two algorithms mentioned above interact with second group of background utilities – graph theory routines.

Graph theory is very important instruments in scheduling theory. Graph, a basic instance of graph theory, is represented by nodes and edges. Relations between those two items describe adjacency matrix. Size of adjacency matrix is equal to number of nodes and value on (i, j) position is equal number of edges from i-th node to j-th node. Graph object from scheduling toolbox is created from adjacency matrix. Adjacency matrix describes only graph structure, but in addition object graph consists of supplement properties with utilities for graph manipulations. Those utilities allow supplement values to the edge, to the node, allow add graphics parameters as node position or a color. For graph manipulation functions as subgraph

cut and graph merge are prepared. Moreover, we prepared some basic graphs algorithms, e.g. successors and predecessors search, or shortest path construction by Floyd's algorithm. One of toolbox routine is graph editor providing comfortable graphics user interface for graph construction. The editor allows adding or removing nodes, edges and their parameters in the graph. Each precedence constrains between tasks are described by the direct acyclic graph, where nodes of graph represents tasks, and edges are precedence constrains. From this point of view we can say that tasksets and graphs can be reciprocal transferable. For this transformation are available functions which allow user defined transfer from graph to taskset and back.

Description of all functions of TORSCHE scheduling toolbox overreaches this paper. Toolbox includes many minor functions which increase usability. Some of them are interfaces functions for integer linear programming solvers or boolean satisfiability solver and their configurations utilities. Next one is XML configuration file which describes all scheduling algorithms and allows interconnect toolbox with web interface or with prepared expert system for automatic algorithms choice. From the above point of view take this paper as preview of news from prepared version 0.1.3 for planed release in February 2006.

References:

- [1] KUTIL, M.: *Scheduling Toolbox First Preview*. Praha: VŠCHT, 2004, pp. 297-303.
- [2] LEUNG, J.: *Handbook of Scheduling. Algorithms, Models, and Performance Analysis*. Chapman and Hall/CRC, 2004.
- [3] ŠŮCHA, P. – POHL, Z. – HANZÁLEK, Z.: *Scheduling of Iterative Algorithms on FPGA with Pipelined Arithmetic Unit*. Los Alamitos: IEEE Computer Society Press, 2004, pp. 404-412.
- [4] DCE CTU FEE: *Scheduling Toolbox – TORSCHE – Manual* Praha: ČVUT, 2006,

This research has been supported by CTU0506513.