# Tabu search for high level synthesis from [3]

Marc Sevaux *

July 26, 2010

## 1 Background and description of the problem

Electronic chip design is of widespread importance throughout the electronics industry, constituting an essential underpinning of computer-based applications. A critical problem in chip design is that of High Level Synthesis (HLS) [2], which consists of minimizing the total cost of an architecture that is required for implementing a particular application in an embedded system (*e.g.* a mp3 decoder device).

A classical way to approach such a problem, which involves discrete sequencing requirements, is by means of an integer linear programming (ILP) model, and this approach has been gaining increased favor with recent advances in the software for solving ILP models. A key finding of our present study is that our tabu search (TS) approach, as described below, is dramatically superior to solving the problem by integer linear programming, which always fails for large real life problem instances. Consequently, we find that our TS method has special merit for practical problems faced in the electronics field.

The High Level Synthesis problem can be model as a particular scheduling problem where the objective, contrary to classical scheduling objectives related to jobs completion, is to minimize the number of resources used for scheduling the jobs in their time windows.

More formally, a list of $n$ jobs is to be scheduled on identical parallel processors, and for each job $i$, a release date $r_i$, a due date $d_i$, a processing time $p_i$ and a set of predecessors $PRED_i$ are given. Let $s_i$ be the starting time of job $i$ that has to be determined. Apart from satisfying precedence constraints, job $i$ cannot start before its release date ($s_i \geq r_i$) and has to be completed before its due date ($s_i + p_i \leq d_i$). The objective is to minimize the number of processors used while respecting all the constraints. This problem is $\mathcal{NP}$-hard.

## 2 Application of Tabu Search for HLS

Sevaux and Sörensen [4] proposed a tabu search method for the problem without precedence constraints. This method begins by computing a lower bound $m$ on the number of processors and then solves the classical $Pm|r_i|\sum U_i$ scheduling problem using tabu search. The tabu search procedure consists of either inserting a late job between two early jobs or replacing a late job with a selected subset of consecutive early jobs. This method uses the sum of processing times of late jobs as a tabu criterion.

The method is designed to handle precedence constraints as well. A preprocessing phase is performed in order to tighten both release dates and due dates based on implications of precedence constraints by computing latest starting times and earliest due dates. Moreover, the method follows a reverse approach in comparison to [4]. From an initial feasible solution with $m$ processors, the number of resources is decreased. The initial feasible solution to the problem with $m$ processors

*Université de Bretagne-Sud, UEB – Lab-STICC – CNRS, Centre de recherche, BP 92116, 56321 Lorient Cedex, France, marc.sevaux@univ-ubs.fr

is obtained by scheduling jobs as early as possible on processors where the remaining idle time on the current processor is smallest (taking into account predecessors).

The method perturbs the solution (removing some processors) to create a set of unallocated jobs. These unallocated jobs are randomly tested for allocation on processors by tabu search.

A reallocation step is undertaken by inserting the jobs on some processors by shifting the position of others (i.e., by rescheduling jobs on the same processor either earlier or later without changing their order). Among many possible insertion positions, a job is inserted at a place where idle time left after insertion is the smallest. After inserting the job, all jobs that are scheduled before it on the same processor are rescheduled so that they can be finished by their earliest completion time and all jobs that are scheduled after it are delayed only by the required minimum amount. This is done following the policy of scheduling the jobs as early as possible.

If it is not possible to insert the job on available processors then the method seeks to reallocate the job by exchanging this job with those consecutively allocated jobs whose schedules overlap with it. By consecutively allocated jobs, we mean those jobs, which are allocated one after the other on the same processor. However, unlike [4], the exchange operation is performed only when the processing time of each consecutively allocated job is less than the unallocated job. Among all possible exchanges, the exchange that leaves the shortest period of idle time is selected. This newly allocated job is assigned a tabu status to prevent it from being removed again for a specific duration.

If a solution with $m$ or less than $m$ processors is obtained then it becomes the new initial solution and the whole process is repeated, otherwise the new solution is discarded and the original solution is perturbed again.

# 3 Successful experiments

Experiments were conducted on instances without precedence constraints to provide a comparison to the TS-SS method from [4]. A large set of instances with precedence constraints were also generated and used for additional experiments. Instances were also taken from the design of real-life embedded systems. Since our work is the first to address the problem as an optimization problem in the electronic community, in order to be able to carry out benchmark comparisons to establish the quality of our outcomes we have developed an integer linear programming (ILP) model and solved it with the GLPK open source software.

On the instances without precedence constraints (from 20 to 100 jobs), our TS-PRED method clearly shows its superiority over the earliest TS-SS procedure. TS-PRED not only returns better quality solutions but is also much faster than TS-SS. Our ILP formulation also proved to be quite effective for these problems, making possible to find all optimal solutions in 600 seconds which was nearly as fast as our TS-PRED method.

On randomly generated instances with precedence constraints (from 25 to 1000 jobs), TS-PRED is able to find almost all optimal solutions when known, while running several times faster than the ILP model.

Results for more challenging real-life instances (from 26 to 4040 jobs) demonstrated a more dramatic superiority for TS-PRED. The ILP model always fails for large instances, while TS-PRED was able to find efficient solutions in 20 seconds on average. Preliminary experiments with a commercial ILP software like Xpress-MP from FICO discloses continued advantages for TS-PRED, where large instances can require more than one hour of running time with the commercial software.

Interest from the electronic community for this kind of approach is increasing based on our finding that results can be obtained very rapidly and can be used in an interactive decision process. Decision makers are currently preparing to integrate this approach in CAD software dedicated to HLS like Gaut [1].

## References


[1] C. Chavet, C. Andriamisaina, P. Coussy, E. Casseau, E. Juin, P. Urard, and E. Martin. A design flow dedicated to multi-mode architectures for dsp applications. In *International Conference on Computer Aided Design 2007 – ICCAD'07*, San Jose, USA, November 2007.

[2] D.D. Gajski, N.D. Dutt, A.C-H. Wu, and S.Y-L. Lin. *High-level synthesis: introduction to chip and system design*. Kluwer Academic Publishers, Norwell, MA, USA, 1992.

[3] M. Sevaux, A. Singh, and A. Rossi. Tabu search for multiprocessor scheduling: application to high level synthesis. *Asia-Pacific Journal of Operational Research*, 2010. To appear.

[4] M. Sevaux and K. Sörensen. A tabu search method for high level synthesis. In *Conférence conjointe Francoro V – Roadef 2007*, pages 395–396, Grenoble, France, 20-23 February 2007.