A Constraint-based Approach to Cyclic Resource-Constrained Scheduling Problem

Alessio Bonfietti
Supervisor: Michela Milano

DEIS, University of Bologna
V.le Risorgimento 2 – 40136 Bologna, Italy
`alessio.bonfietti@unibo.it`

The cyclic scheduling problem concerns assigning start times to a set of activities, to be indefinitely repeated, subject to precedence and resource constraints. It can be found in many application areas. For instance, it arises in compiler design implementing loops on parallel architecture[5], and on data-flow computations in embedded applications[2]. Moreover, cyclic scheduling can be found in mass production, such as cyclic shop or Hoist scheduling problems[6].

In cyclic scheduling often the notion of optimality is related to the period of the schedule. A minimal period corresponds to the highest number of activities carried out on average over a large time window.

Optimal cyclic schedulers are lately in great demand, as streaming paradigms are gaining momentum across a wide spectrum of computing platforms, ranging from multi-media encoding and decoding in mobile and consumer devices, to advanced packet processing in network appliances, to high-quality rendering in game consoles. In stream computing, an application can be abstracted as a set of tasks that have to be performed on incoming items (frames) of a data stream. A typical example is video decoding, where a compressed video stream has to be expanded and rendered. As video compression exploits temporal correlation between successive frames, decoding is not pure process-and-forward and computation on the current frame depends on the previously decoded frame. These dependencies must be taken into account in the scheduling model. In embedded computing contexts, resource constraints (computational units and buffer storage) imposed by the underlying hardware platforms are of great importance. In addition, the computational effort which can be spent to compute an optimal schedule is often limited by cost and time-to-market considerations.

My research focuses on scheduling periodic application. In particular, in first place I have worked together with my research group on a Constraint Programming approach based on modular arithmetic for computing minimum-period resource-constrained cyclic schedules [3]. The solver has several interesting characteristics: it deals effectively with temporal and resource constraints, it computes very high quality solutions in a short time, but it can also be pushed to run complete search. An extensive experimental evaluation on a number of non-trivial synthetic instances and on a set of realistic industrial instances gave promising results compared with a state-of-the art ILP-based (Integer Linear Programming)[1] scheduler and the Swing Modulo Scheduling (SMS)[7] heuristic technique. The main innovation of our approach is that while classical *modular* approaches fix the modulus and solve the corresponding (non periodic) scheduling problem, in our technique the bounds for the modulus variables are inferred from the activity and iteration variables.

The main drawback of this first approach is the underlying hypothesis that the end times of all activities should be assigned within the modulus. Thanks to this assumption, we can reuse traditional resource constraints and filtering algorithms. However the solution quality can be improved by relaxing this hypothesis.

Therefore, we proposed [4] a Global Cyclic Cumulative Constraint (GCCC) that indeed relaxes this hypothesis. We have to schedule all the start times within the modulus λ, but we have no restriction on end times. The resulting problem is far more complicated, as enlarging the modulus produces a reduction of the modular end time of the activities. Figure 1 explains the concept. Suppose the grey activity requires one unit of a resource of capacity 3. If the modulus value is D, then the activity can be scheduled as usual. If the modulus is reduced to C, the starting time of the activity is the same, while the "modular end time" is c and the resource consumption is 2 between 0 and c. If the modulus is further reduced to B the modular end time increases to b. Finally, if the modulus is reduced to A, the modular end point becomes a and the resource consumption is 3 between 0 and a.

In [4] we show the advantages in terms of solution quality w.r.t. our previous approach that was already outperforming state of the art techniques. The experiments highlight that our approach obtains considerably better results in terms of solution quality for high capacity values. Moreover, the results show that, working with acyclic graphs, the GCCC approach obtains an approximately constant resource idle time.

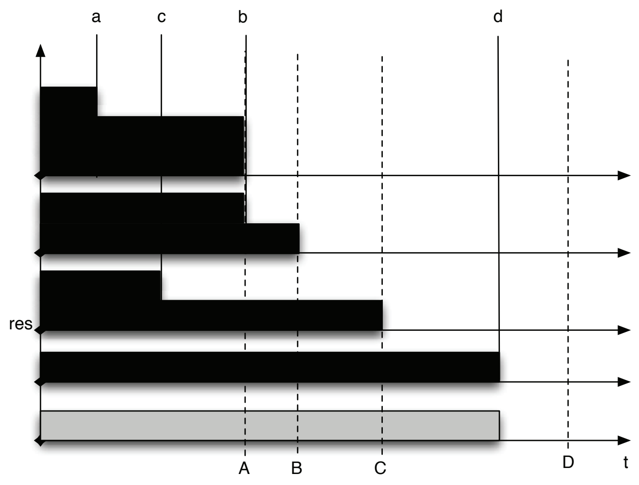

Fig. 1. Resource Profiles w.r.t different modulus values

Further investigation will be devoted to the design of cyclic scheduling heuristic algorithms and their comparison with complete approaches.

References

1. Ayala, M., Artigues, C.: On integer linear programming formulations for the resource-constrained modulo scheduling problem (2010)
2. Bhattacharyya, S.S., Sriram, S.: Embedded Multiprocessors - Scheduling and Synchronization (Signal Processing and Communications) (2nd Edition). CRC Press (2009)
3. Bonfietti, A., Lombardi, M., Benini, L., Milano, M.: A Constraint Based Approach to Cyclic RCPSP. In: CP2011. pp. 130–144 (2011)
4. Bonfietti, A., Lombardi, M., Benini, L., Milano, M.: Global cyclic cumulative constraint. In: Beldiceanu, N., Jussien, N., Pinson, E. (eds.) 9th International Conference on Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems (CPAIOR'12). Lectures Notes in Computer Science, vol. 7298, pp. 81–96. Springer Verlag, Nantes, France (Jun 2012)
5. Dupont de Dinechin, B.: From Machine Scheduling to VLIW Instruction Scheduling (2004)
6. Hanen, C.: Study of a NP-hard cyclic scheduling problem: The recurrent job-shop. European Journal of Operational Research 72(1), 82–101 (1994), http://www.sciencedirect.com/science/article/B6VCT-48NBGY3-226/2/e869b267f2deeef5b90a18f742a2e2c4
7. Llosa, J., Gonzalez, A., Ayguade, E., Valero, M.: Swing modulo scheduling: A lifetime-sensitive approach. In: pact. pp. 80–87. Published by the IEEE Computer Society (1996), http://en.scientificcommons.org/43187025http://www.computer.org/portal/web/csdl/doi/10.1109/PACT.1996.554030