Modulo Scheduling with Regular Unwinding

Benoît Dupont de Dinechin

March 30, 2004

We introduce a new technique for modulo scheduling, based on the unwinding of the modulo scheduling problem, and the acyclic scheduling of the unwinded problem under an additional constraint of regularity. Given λ the modulo schedule initiation interval, a regular unwinded schedule is such that two successive instances of any operation are scheduled at least λ cycles apart. For a given λ, we establish the equivalence between the modulo schedules, and the regular unwinded schedules of suitable size.

A main benefit of the regular unwinding technique is the re-formulation of the modulo scheduling problems in the classic framework of acyclic scheduling. In particular, we introduce new modulo scheduling problem relaxations that are solvable in pseudo-polynomial time. These results are obtained by combining regular unwinding with the time-constrained instruction scheduling relaxation of Leung, Palem & Pnueli [2].

Modulo scheduling [3, 1] is an instruction scheduling technique used for software pipelining inner program loops. In modulo scheduling problems, a set of operations $\{O_i\}_{1 \le i \le n}$ is repeatedly executed with a period of λ cycles, the *initiation interval*. Precisely, we denote $\{\sigma_i\}_{1 \le i \le n}$ the schedule dates, and the execution is constrained as follows:

- Uniform dependence constraints denoted $O_i \xrightarrow{\alpha_i^j, \beta_i^j} O_j$: for each such dependence, a valid modulo schedule satisfies $\sigma_i + \alpha_i^j - \lambda\beta_i^j \le \sigma_j$. The *latency* α_i^j and the *distance* β_i^j of the dependences are non negative integers. The *carried* dependences are such that $\beta_i^j > 0$.

- Modulo resource constraints: each operation O_i requires $\vec{b}_i \ge \vec{0}$ cumulative resources for all the time intervals $[\sigma_i + k\lambda, \sigma_i + k\lambda + p_i - 1], k \in \mathbb{Z}$, and the total resource use at any time must not exceed $\vec{B}$. The positive integer value p_i is the processing time of operation O_i.

References

[1] M. Lam: *Software Pipelining: an Effective Scheduling Technique for VLIW Machines* SIGPLAN Conference on Programming Language design and Implementation – PLDI'88, June 1988.

[2] A. Leung, K. V. Palem, A. Pnueli: *Scheduling Time-Constrained Instructions on Pipelined Processors* ACM TOPLAS Vol. 23, No. 1, Jan. 2001.

[3] B. R. Rau , C. D. Glaeser: *Some Scheduling Techniques and an Easily Schedulable Horizontal Architecture for High Performance Scientific Computing* 14th Annual Microprogramming Workshop on Microprogramming – MICRO-14, Dec. 1981.