Synthesis of Software Kernels in Hardware

Vitaly Nikolyenko
vnik5287@uni.sydney.edu.au

Abstract

Applications typically exhibit vastly different performance characteristics, requiring various amounts of computational resources. Despite this application diversity, hardware-to-software, i.e., designing applications for a specific *fixed* architecture, remains to be the most common approach in software construction.

On the contrary, reconfigurable computing allows for a software-to-hardware approach, i.e., designing an architecture for a specific task. This approach targets application requirements instead, in order to produce an optimal domain-specific solution. Furthermore, substantial benefits often can be achieved by reimplementing a software's critical regions (or critical kernels) as a custom reconfigurable circuit coprocessor.

However, efficient hardware/software co-synthesis still remains a major challenge in reconfigurable computing. An efficient scheduling algorithm is required to get close to performance offered by fixed application-specific custom circuits.

Scheduling alone is an NP-hard problem. Furthermore, massively parallel architectures implementing local register files add extra complexity, since cross-links (i.e., communication channels between processing elements) must be considered when performing schedule allocations.

An optimal schedule can be found by expressing the scheduling problem as a multi-objective function of reducing the makespan and minimising the required number of cross-links. Integer linear programming (ILP) can then be used for most given problem instances to produce and optimal schedule considering the cross-links constraints. Of course, this mathematical program is not practical for large problem sizes but it is a good yardstick to compare how good available heuristics perform in comparison to the optimal solution.