Mapping dynamic applications on MPSoC systems

Zubair Wadood Bhatti, Davy Preuveneers, Roel Wuyts, Yolande Berbers.
Distrinet, Katholieke Universiteit Leuven, Belgium.
{zubairwadood.bhatti, davy.preuveneers, roel.wuyts, yolande.berbers}@cs.kuleuven.be

My PhD research will investigate the optimization of executing dynamic applications on heterogeneous multi-processor systems-on-chips (MPSoCs). The optimization is aimed at minimizing the resource requirements through efficient deployment of software onto hardware. The deployment will be carried out by run-time environment assisted by input from a pre-deployment exploration tool chain. The targeted application's resource requirements cannot be predicted beforehand due to dependences on user input and application data. On one hand a pure design time approach to the allocation and scheduling parts of applications to different cores and memory hierarchies will have to stick to the worst case estimates and will therefore result in a suboptimal resource usage. On the other hand a pure run-time approaches either has a very high overhead penalty [1] or do resource management at a very abstract level with a high granularity and a sub-optimal performance.

In our research we assume that the application is represented as an annotated distributed model of computation such as the grey-box model [2]. The grey-box model can be described as a fine grained Khan Process Network (KPN) annotated with information such as the time required and energy consumed for running a process on a specific processing element. The MPSoC platform can be expressed as a Network-on-Chip such that the optimization process can be formulated as a problem of mapping one network onto another i.e. the allocation and scheduling of the application processes onto processing elements and buffers onto the memory hierarchy.

The resources we identify as being the most important in the context of MPSoC systems are energy, memory, on-chip communication and processing elements. There are several approaches that try to simplify the problem by decomposing it into sub-problems of optimization of different types of resources or only deal with a subset of the whole problem for example [4] provides a technique for optimizing memory while meeting execution time requirements. Due to the existence of a large amount of dependencies between the usages of all these resources, such simplifications may be insufficient for providing optimal solutions with the predicted in the increase complexity of applications and platforms in the near future. For example, on optimal of usage processing elements would require application processes to be distributed all over the whole platform as much as possible so that the load is well balanced allowing processors to run at lower frequencies. However, such an approach would put more pressure on the communication channels. Similar tradeoffs exist between context switches, throughput (for streaming applications) and memory buffer requirements. Therefore a local optimal solution for one particular type of resource does not necessarily correspond to the global optimal solution. At the end of the day we are confronted with the extremely difficult task of evaluating the relative importance of one resource to another i.e. deciding whether it is good to save 20% of memory by giving up 15% of the communication resources and so on.

We propose a simplification of the problem by explicit differentiation between exhaustible and inexhaustible resources, treating exhaustible resources as costs and the inexhaustible ones as constraints. For this optimization problem we consider the total energy that will be consumed by the system for a given mapping solution to be the one and only cost of the solution that needs to be minimized. Everything else, for example the storage, processing or communication resources used by the application just need to satisfy certain constraints This implies that our objective is not to minimize the usage of any of these resources, but rather that the total energy consumed by the system is minimized. Our approach however has significant challenges when multiple applications are running on the same platform. For such a situation we propose to extend the system scenario aware execution [3] method to allow applications to put resource constraints on each other at runtime, in a fashion similar to how a dynamically changing environment can put performance constraints on the application at runtime and cause the resource requirements to change.

At design time our tools will do exploration in a four dimensional space {Computation, Communication, Memory and Energy} where each point represents a possible mapping solution. The exploration will result in a Pareto optimal surface bounded by very relaxed constraints such as the total allocatable memory, communication and processing resources for the application. At runtime the search space is limited to the Pareto optimal surface this time bounded by the more restrictive constraints imposed by the runtime system scenario.

Some basic design time exploration tools are available, but since the computational complexity of exploration increases exponentially with the size and complexity of the application and of the platform, so there is a need to find efficient search heuristics. At the same time we need to refine the cost models, such as the energy consumed by the memory hierarchy. More accurate cost models will help us relax some conditions and find better solutions. For the runtime phase we need a framework for negotiating resource requirements between applications by predicting energy costs of executing applications under different sets of resource constraints and using these costs to minimize the total energy consumption of the system, while still meeting performance requirements of all the applications.

References:

1. **Yang, Peng, et al.** *Managing dynamic concurrent tasks in embedded real-time multimedia systems.* Kyoto : ACM, 2002. International Symposium on Systems Synthesis. pp. 112 - 119.

2. **Thoen, Filip and Catthoor, Francky.** *Modeling, Verification, and Exploration of Task-Level Concurrency of Real-Time Embedded Systems.* s.l. : Kluwer Academic Publishers, 2000.

3. **Gheorghita, Stefan Valentin, et al.** *System-scenario-based design of dynamic embedded systems.* ACM, 2009, Transactions on Design Automation of Electronic Systems, Vol. 14. ISSN:1084-4309.

4. **Heikki Orsila, et al.** *Automated memory-aware application distribution for Multi-processor System-on-Chip* 2007 Elsevier B.V. Journal of Systems Architechture