RESEARCH PAPERS
FACULTY OF MATERIALS SCIENCE AND TECHNOLOGY IN TRNAVA
SLOVAK UNIVERSITY OF TECHNOLOGY IN BRATISLAVA

10.2478/rput-2014-0005 2014, Volume 22, Special Number

TOWARDS A COMPLETE MODEL FOR SOFTWARE COMPONENT DEPLOYMENT ON HETEROGENEOUS PLATFORM

Ivan ŠVOGOR[1]

ABSTRACT

This report briefly describes an ongoing research related to optimization of allocating software components to heterogeneous computing platform (which includes CPU, GPU and FPGA). Research goal is also presented, along with current hot topics of the research area, related research teams, and finally results and contribution of my research. It involves mathematical modelling which results in goal function, optimization method which finds a sub-optimal solution to the goal function and a software modeling tool which enables graphical representation of the problem at hand and help developers determine component placement in the system design phase.

KEY WORDS

Heterogeneous computing, software components, optimization

INTRODUCTION

Research topic and research goal

The research topic presented in this report is derived from the SSF[2] RALF3 project (1), currently undergoing at the Mälardalen University, Sweden. The project focuses on developing theories, technologies and tools which support design and development of embedded systems that process large amount of data in real-time using heterogeneous hardware. The project targets automotive industry, 3D sensing, space, medicine, and other data intensive environments.

My research within the project is related to research question C1: "*How can information intense embedded systems best utilize a specific heterogeneous platform?*" and with a research challenge R3: "*How can a developer be helped in complex task of allocating components to computing units in a cost- and power-effective way? - Among other things, this requires means to obtain relevant system-level quality properties for a particular allocation, from EFPs[3] of individual components*".

Heterogeneous hardware consist of different computing unit types, where each unit can be dedicated for a particular type of computation. Using different computational units is already a

[1] Ivan ŠVOGOR
University of Zagreb, Faculty of organization and Informatics, Pavlinska 2, 42000 Varaždin, Croatia,
isvogor@foi.hr

[2] Swedish Foundation for Strategic Research
[3] Extra-functional properties

proved approach in high performance computer systems, in which GPUs process highly parallel computation, and in which cores from multicore CPUs perform different tasks in parallel. This is also becoming of significant importance in embedded systems where such hardware enables processing of large amount of data streams in real-time. This great potential for increased performance is still not fully utilized due to a lack of software methods for an efficient and optimal placement (allocation) of software to the heterogeneous platform by which an optimal, or sufficiently good performance is obtained.

Different software allocations result with different performance and it is not obvious which allocation would enable the best performance. In addition, best allocation candidate might not be allowed due to different constraints; this can be due to limitation of resources of a particular unit (such as memory or communication capacity or energy consumption), or due to some architectural decisions related to specific requirements (such as a requirement that two components are not allowed to be allocated to the same physical computational unit). A "*trial and error*" method by repeated allocations and then measurements is an inefficient procedure, in particular when the software implementation may depend on which computational unit type will be executed. For this reason, an allocation method is desired in an early development phase.

Previous research

Research groups and lead researchers of the area: a*) Mälardalen University, MRTC Group - I. Crnkovic*, b) *Carnegie Mellon University, SEI - K. Wallnau*, c) *Karlsruhe University, IPD - R. Reussner*, d) *Microsoft Research, C. Szysperski*, e) *Politehnico di Milano, DEEPSE R. Mirandola*, f) *University of Padenborn, C. Heinzemann, S. Becker*, g) *University of Southern California, SoftArch team, N. Medvidovic, etc*.

According to "*Trends in Software Engineering*" from Liggesmeyer (2) special attention in software engineering research is brought to modeling tools which integrate whole development cycle including quality assurance, testing, reliability analysis, model based safety and model based analysis. Moreover, related work can be categorized in following categories: a) *software modeling*, b) *software architecture*, c) *modeling heterogeneous platforms*, d) *FPGA, GPU, CPU communication middleware*.

The research presented in this report focuses on a), b) and c) while d) is out of scope. Considering this scope, publications mostly relate to component-oriented frameworks for software architecture modeling which enable reasoning about extra-functional properties (e.g. Palladio component model (3), ProCom component model (4). Also, worst-case execution time analysis is a popular subject since execution time is an EFP which contributes at most to the real-time characteristics of the system (5, 6).

However, not a lot of work addresses component-oriented frameworks targeted for heterogeneous platform, and specifically allocating software components to heterogeneous computational units. Several works relate to tasks allocation to different processing units with some resource constraints and to searching for an optimal load balancing across the system (7), (8) or a good average-case performance (9). In (10), a dynamic reallocation is enabled in combination with performance monitoring. More about this topic and problems related to heterogeneous platforms and challenges in components synchronization between the platforms can be found in (11).

RESEARCH APPROACH

Mathematical model

We defined a mathematical model in the form of a cost function which evaluates allocations, therefore pro- viding a normalized value of resource usage costs. The cost function considers different system resources, communication and predefined user constraints. Using this cost function one can compare different allocations and evaluate which one is most suitable for the target platform.

Also the issue related to different measurement units (e.g. execution time is expressed in milliseconds, memory in megabytes, energy consumption in watts per hour, etc.; all those should be comparable) is solved using AHP - a common method for complex multidimensional choices, alternatives and tradeoffs.

Optimization method

A problem emerges from the size number of all possible allocations (which be executed in polynomial time, i.e. if we have m hardware units and n software components, using permutation with repetition one can easily calculate the search space as m^n). Small increases of either com- putting units or software components cause enlargement of the search space, which is not searchable in a polynomial time.

Mathematical model provides a cost function which can compare different software allocations, but considering the search space, some optimization is necessary. In current research we used Genetic Algorithm to evolve the best allocation (instead of exhaustive search).

Software modelling

To enable automatic allocation on software components on heterogeneous computing units one needs to model the software architecture and provide all the necessary information (model constraints) required for the mathematical model. This input will be through a graphical tool which uses EMF based metamodel for modeling component oriented software requirements and heterogeneous platforms upon which it should be allocated.

RESEARCH RESULTS AND DISCUSSION

As this research is a work in progress, I present current insights and research contribution along with future development.

Current insights

We proposed an extension of our previous model (12) for optimization of component allocation on a heterogeneous embedded platform. We improved the model by addressing the following: a) handling different measurement units, b) handling the subjective judgment of the criteria for allocation decision with AHP consistency index verification, c) including the bandwidth constraint for the communication and d) enabled architect defined constraints. The solution[4] provides a semi-optimal allocation model which uses a Genetic Algorithm (any other optimization technique can be applied) and Analytical Hierarchical Process (for handling user preference biasness – e.g. to determine importance of resources). Also in our previous research

[4] Current findings are published in International Conference on Information Technology Interfaces (IEEE) and Journal of Computing and Information Technology (SCOPUS, INSPEC, CSI).

we present an example in the form of component software architecture for an Autonomous Underwater Vehicle (AUV). AUV has a heterogeneous computing platform which consists of multicore CPUs, GPU and FPGA boards. Besides naval research, this finds application in automotive industry, space and warfare research.

Future work

Current method enables efficient placement of software components on computational units of a heterogeneous platform. It considers multiple criteria which are both system defined and architect defined. The result is a semi-optimal component allocation for a particular system calculated using input matrices with following vales: a) resource consumption requirements, resource availability, b) platform communication cost, c) bandwidth, d) communication intensity. There are some aspects of the research which require further investigation; a) communication intensity, b) EFPs and c) modeling tool.

Most of the values can be acquired by measurement, calculation or empirically, however communication intensity needs further discussion and research because it cannot be easily quantified. Its intent is to envelop frequency of communication between components so one can quantify the usage of channels between them. One way to look at it is the number of function calls, channel data type i.e. signal data or streaming data, or the approach which I used in the latest paper; approximation with AHP scale.

One also must consider non-functional constraints, e.g. development effort. In real world some components require great development efforts to be implemented on certain platforms. As for the modeling tool, I am currently developing an Eclipse Plugin (EMF[5], GMF[6] and GEF[7] based) which allows automatic component allocation in early architecture design phase. This is an ongoing effort for the ASE conference.

CONCLUSION

The current model is works well, however it requires refinement and real-world verification. Along with the theory, this research considers its verification using a robot which features heterogeneous computing platform. An experiment will be designed to verify how does calculated software allocation reflect the real world use (i.e. create allocations optimized for different scenarios, e.g. minimize energy consumption).

REFERENCES

1. R. P. Web, 2013, http://www.mrtc.mdh.se/projects/ralf3/, [Accessed: Jan 2013].
2. LIGGESMEYER, P. and TRAPP, M. 2009. Trends in embedded software engineering. *IEEE Software*, **26**(3).
3. BECKER, S., KOZIOLEK, H. and REUSSNER, R. 2009. *Model–based performance prediction with the palladio component model*.
4. SENTILLES, S., VULGARAKIS, A., BUREŠ, T., CARLSON, J. and CRNKOVIC ,I. 2008. *A component model for control–intensive distributed embedded systems*, pp. 310–317.
5. CARLSON, J., FELJANMÄKI-TURJA, J. and SJÖDIN, M. 2010. *Deployment modelling and synthesis in a component model for distributed embedded systems*, pp. 74 82.

[5] Eclipse Modeling Framework
[6] Graphical Modeling Framework
[7] Graphical Editing Framework

6. FREDRIKSSON, J. SANDSTRÖM, K. and AKERHOLM, M. 2005. *Optimizing resource usage in component–based real- time systems*, pp. 49–65.
7. RISTAU, B., LIMBERG, T. and FETTWEIS, G. 2008. *A Mapping Framework for Guided Design Space Exploration of Heterogeneous MP-SoCs*, pp. 780–783.
8. WANG, S., MERRICK, J. R. and K. SHIN, G. 2004. *Component allocation with multiple resource constraints for large embedded real–time software design*, pp. 219–226.
9. FELJAN, J., CARLSON, J. and SECELEANU, T. 2012.*Towards a model-based approach for allocating tasks to multicore processors*, pp. 117–124.
10. MALEK, S., MEDVIDOVIC, N. and MIKIC-RAKIC, M. 2012. An extensible framework for improving a distributed software system's deployment architecture. *IEEE Transactions on Software Engineering*, **38**(1), pp. 73–100.
11. SENOUCI, B. 2008. *Multi–cpu/fpga platform based heterogeneous multiprocessor prototyping: New challenges for embedded software designers*, pp. 41–47.
12. ŠVOGOR, I. CRNKOVIC, I. and VRCEK, N. 2013. Multi-criteria software component allocation on a heterogeneous platform. In: International Conference on Information Technology Interfaces. Zagreb, pp. 341–346.