R&D Challenges and Emerging Solutions for Multicore Deployment/Configuration Optimization

Jules White
Bradley Department of Electrical and Computer Engineering
Virginia Tech, Blacksburg, VA, USA
Email: julesw@vt.edu

Douglas C. Schmidt
Department of Electrical Engineering and Computer Science
Vanderbilt University, Nashville, TN, USA
Email: schmidt@dre.vanderbilt.edu

Categories and Subject Descriptors

D.2.11 [**Software**]: Software Engineering—*Software Architectures*

General Terms

Design

Keywords

deployment optimization, multicore processors

1. INTRODUCTION

Emerging trends and opportunities. The most recent trends in distributed real-time and embedded systems (DRE) development have begun decoupling software from the underlying hardware. For example, the Software Communications Architecture (SCA) provides a standard operating environment that allows software to be ported across hardware platforms. The AUTOSAR standard is an open standardization effort developed by 35 different automobile manufacturers, tool vendors, and suppliers that provides standardized hardware abstractions and software interfaces that allow software functionality to be moved between automotive ECUs with minimal effort. The same decoupling has also taken place in other domains, such as the aeronautics industry.

Leveraging this decoupling to optimize deployment and configuration is a critical concern in DRE systems that run on multicore processors. Current processors are increasing performance by adding processing cores rather than increasing clock speed. To reap the benefits of leveraging additional cores, the deployment topology must be carefully optimized to ensure that the software is distributed across cores in a manner that maximizes performance. For example, if a deployment topology can be found that uses fewer processors in a flight avionics system, each pound of processor savings results in $200 in decreased fuel costs and a decrease in greenhouse gas production from less burned fuel [9]. In previous generation systems with single core processors or where the software was tightly-coupled to the hardware, deployment and configuration was less critical.

Although this decoupling trend and multicore processors have introduced a number of complexities, embedded system manufacturers have embraced this decoupling to help utilize hardware more efficiently. Decoupling software from specific hardware processors allows software components to be flexibly deployed and configured to optimize the utilization of the hardware. The deployment topology (the mapping of software to hardware) or software configuration can be optimized to decrease the total hardware processors required, improve performance, decrease power consumption, lower heat dissipation, or reduce cost.

Open problems. Determining how to deploy and configure software in multicore DRE systems is a challenging problem [6] due to the large number of complex constraints that must be dealt with. Developers must ensure that each software component is provided sufficient processing time on a core to meet any real-time scheduling constraints. Resource constraints, such as total available memory or cache on each core or processor, must also be respected by the mapping of software to hardware. Finally, components can have complex placement or co-location constraints, such as requiring specific software components to be deployed to cores on the same processor or processors a minimum distance from the perimeter of an enclosure to increase survivability when dropped.

2. DEPLOYMENT OPTIMIZATION CHALLENGES

While deployment optimization offers a number of benefits, developers of systems running on multicore systems face a daunting series of conflicting constraints and optimization goals when determining how to deploy software to hardware. For example, it is hard to find a valid solution for a single deployment constraint, such as ensuring that all of software tasks can be scheduled to meet real-time deadlines, in isolation, using conventional techniques, such as bin-packing. It is even harder, moreover, to find a valid solution when considering many deployment constraints, such as satisfying resource requirements of software tasks in addition to ensure schedulability. Optimizing the deployment topology of a system to minimize consumed network bandwidth or other dynamic properties is harder still since communication between software tasks must be taken into account, instead of simply considering each software task as an independent entity.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
FoSER 2010, November 7–8, 2010, Santa Fe, New Mexico, USA.
Copyright 2010 ACM 978-1-4503-0427-6/10/11 ...$10.00.

2.1 Challenge 1: Satisfying Complex Deployment Constraints

The deployment topology of a system determines how resources are allocated to different system modules and can have a dramatic impact on the performance and other non-functional characteristics of the system. For example, if multiple CPU intensive components are mapped to the same processor or core, the system may exhibit substantially worse performance than if these same modules are distributed across a group of cores or processors. In other situations, the communication overhead of distributing a set of intercommunicating software modules across multiple processors or cores may have a more adverse performance impact than co-locating the modules.

Developers of systems that run on multicore processors must carefully analyze the complex tradeoffs and implications of different deployment topologies to ensure that they meet the non-functional requirements of the system. In many cases, real-time scheduling constraints must be guaranteed across multiple processors and cores. Additional constraints, such as co-location constraints limiting the mapping of a component and its fault-tolerant replica to the same core, must also be adhered to. Finding deployments that meet a complex series of deployment constraints is difficult [1, 4].

2.2 Challenge 2: Wide Variation in Deployment Constraints

Further exacerbating the complexity of finding a deployment topology that satisfies a series of constraints is that the constraints vary significantly across systems. In some systems, such as flight avionics, real-time scheduling and communication bandwidth are the dominant constraints. In other domains, such as cloud computing, request processing latency and fault-tolerance are more critical.

Because the constraints vary across domains and a wide variety of constraints are present, no single algorithm can be applied. For example, heuristic bin-packing can be used to derive deployment topologies that meet real-time scheduling constraints. However, these same algorithms do not work when co-location constraints are included. The varying mixes of constraints make it difficult to produce generalized solutions that can be employed across multiple domains.

2.3 Challenge 3: Deployment Optimization

In many cases, finding a valid deployment topology is not sufficient. Often, deployments are sought that optimize specific properties of the mapping of software to hardware. For example, system designers may seek to optimize the deployment topology of a legacy system to reduce the number of processors used by the current software and make way for software enhancements.

Constraint logic programming and other exhaustive optimization techniques are often used to derive optimized designs. Deployment problems, however, often have complex real-time scheduling and resource constraints that cause these techniques to scale poorly. Finding flexible deployment optimization algorithms that can handle a wide variety of constraint types and provide flexible design optimization is challenging.

3. EMERGING RESEARCH SOLUTIONS ⇒ MODEL-DRIVEN DEPLOYMENT/ CONFIGURATION OPTIMIZATION

A promising approach to automating and optimizing the deployment/configuration of software in multicore systems is to apply a combination of Model-Driven Engineering (MDE) and heuristic/-metaheuristic optimization algorithms. This approach allows system developers to use high-level visual models of the software and hardware platform to capture key deployment/configuration constraints and optimization goals. Heuristic/-metaheuristic optimization algorithms are then used to automatically derive optimized deployment/ configuration plans for the software. An example MDE deployment tool and automation process is shown in Figure 1.

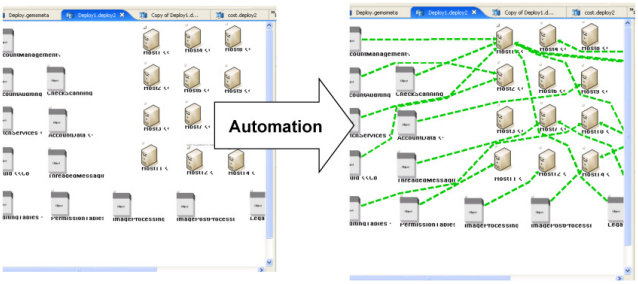

Figure 1: Model-Driven Deployment and Configuration Optimization Tools

Deployment and Configuration Modeling. Deploying and configuring component-based software, such as SCA components, exposes developers to a vast number of complexities related to the scheduling, power consumption, co-location, and memory consumption. Model-driven engineering helps to reduce the complexity of deployment and configuration by using carefully designed abstractions to shield developers from unnecessary deployment/configuration details. Model-driven deployment/configuration techniques use high-level domain-specific modeling languages, that provide language abstractions specifically tailored to the terminology and notations of domain experts.

Once deployment/configuration models are constructed, they can be used as input to a variety of code generators and optimization algorithms. For example, XML deployment descriptors for Corba Component Model (CCM) or Software Defined Radio (SDR) applications can be generated from the models. A key benefit of using an MDE approach is that the deployment/configuration models can be rapidly refactored and the implementation artifacts, such as XML descriptors, can be regenerated automatically. Moreover, implementation artifacts for multiple target languages/frameworks can be generated from a single model. In past collaborations with BAE Systems, PrismTech, IBM, and Siemens, we have developed deployment/configuration modeling tools and techniques for SDR, CCM, EJB, CORBA, and smartphone software [11, 12].

Heuristic/-metaheuristic Deployment and Configuration Optimization. Automation and optimization is an important component of reducing the complexity of deploying and configuring software on multicore processors. Deployment and configuration problems have extremely large solution spaces with complex constraints, which makes man-

ually finding solutions difficult. Because constraint logic programming and other exact optimization techniques do not scale well, heuristic and metaheuristic algorithms [5, 8], such as first-fit bin-packing [3, 7, 2] and genetic algorithms, are commonly employed.

Although heuristic algorithms scale well, they are often tightly coupled to a specific mix of constraints and often do not produce solutions that meet all desired constraints. These algorithms also rarely allow for customizable objective functions to be defined. These algorithms work well on the narrow problem definitions that they are designed for.

Metaheuristic algorithms can operate on a wide variety of customizable objective functions, making them ideal for flexible optimization. These algorithms, however, do not work well in tightly constrained search spaces, such as those typical of deployment problems.

To overcome the limitations of applying either type of algorithm independently, hybrid approaches can be used that apply the algorithms in tandem. For example, a metaheuristic algorithm, such as particle swarm optimization, can be used to drive a bin-packing algorithm to produce deployment topologies. These combined algorithmic approaches typically exhibit the best attributes of each individual algorithm.

4. CONCLUDING REMARKS

In prior work, we have developed a number of model-driven deployment and configuration tools based on the Generic Eclipse Modeling System (GEMS). GEMS is a graphical modeling platform that we have developed that has been incorporated into the Eclipse Foundation and is distributed through its worldwide mirrors (`http://eclipse.org/gmt/gems`). Developers use these tools to build visual models of software, target hardware, and deployment/configuration constraints.

We have alo developed a suite of automated deployment and configuration algorithms, called the *ASCENT Design Studio* (ASCENT) (`http://code.google.com/p/ascent-design-studio`). ASCENT provides developers with a number of algorithms that can be plugged into a model-driven deployment/configuration process for: 1) deploying software to hardware processing cores to minimize network communication, power consumption, and cost [6]; 2) optimizing software configuration to meet available resource levels on a target hardware platform; 3) co-design of software and hardware configurations with strict resource constraints [10]; and 4) automatically identifying and correcting deployment/configuration errors. The ASCENT algorithms provide pluggable interfaces that can be adapted for new optimization goals and constraints, such as heat dissipation. In current work with Lockheed Martin, we are developing new plug-ins to optimize multicore deployment topologies to exploit beneficial cache effects.

In collaboration with Lockheed Martin, Siemens, Raytheon, and others, we have demonstrated that these techniques can improve deployment/configuration in a number of domains, such as: mapping medical imaging algorithmic steps to cores in CELL processors, software to ECUs in automobiles, avionics pub/sub components to processors in planes, and CORBA components to hosts. In the avionics domain, our empirical results showed that the deployment and configuraiton optimizations were able to reduce the total required processors by 25% and the total network bandwidth consumed by over 33%. Results from other domains, such as deployment within datacenters and for wireless sensor networks has shown power consumption reductions of 70-1500% versus traditional bin-packing based software deployment approaches [6].

5. REFERENCES

[1] H. Beitollahi and G. Deconinck. Fault-Tolerant Partitioning Scheduling Algorithms in Real-Time Multiprocessor Systems. *Pacific Rim International Symposium on Dependable Computing, IEEE*, 0:296–304, 2006.

[2] A. Bertossi, L. Mancini, and F. Rossini. Fault-Tolerant Rate-Monotonic First-Fit Scheduling in Hard-Real-Time Systems. *IEEE Transactions On Parallel and Distributed Systems*, pages 934–945, 1999.

[3] A. Burchard, J. Liebeherr, Y. Oh, and S. Son. New Strategies for Assigning Real-time Tasks to Multiprocessor Systems. *IEEE Transactions on Computers*, 44(12):1429–1442, 1995.

[4] A. Carzaniga, A. Fuggetta, S. Richard, D. Heimbigner, A. van der Hoek, A. Wolf, and COLORADO STATE UNIV FORT COLLINS DEPT OF COMPUTER SCIENCE. *A Characterization Framework for Software Deployment Technologies*. Defense Technical Information Center, 1998.

[5] C. Fonseca, P. Fleming, et al. Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization. In *Proceedings of the fifth international conference on genetic algorithms*, pages 416–423. Citeseer, 1993.

[6] Jules White and Brian Dougherty and Chris Thompson and Douglas C. Schmidt. ScatterD: Spatial Deployment Optimization with Hybrid Heuristic / Evolutionary Algorithms.

[7] S. Lauzac, R. Melhem, and D. Mosse. Comparison of Global and Partitioning Schemes for Scheduling Rate Monotonic Tasks on a Multiprocessor. In *10th Euromicro Workshop on Real Time Systems*, pages 188–195, 1998.

[8] R. Poli, J. Kennedy, and T. Blackwell. Particle swarm optimization. *Swarm Intelligence*, 1(1):33–57, 2007.

[9] N. R. C. Steering Committee for the Decadal Survey of Civil Aeronautics. *Decadal Survey of Civil Aeronautics: Foundation for the Future*. The National Academies Press, 2996.

[10] J. White, B. Doughtery, and D. C. Schmidt. ASCENT: An Algorithmic Technique for Designing Hardware and Software in Tandem. *IEEE Transactions on Software Engineering Special Issue on Search-based Software Engineering*, (to appear).

[11] J. White, A. Nechypurenko, E. Wuchner, and D. C. Schmidt. Automating Product-Line Variant Selection for Mobile Devices. In *Proceedings of the 11th Annual Software Product Line Conference (SPLC)*, Kyoto, Japan, Sept. 2007.

[12] J. White and D. C. Schmidt. Automated Configuration of Component-based Distributed Real-time and Embedded Systems from Feature Models. In *Proceedings of the 17th Annual Conference of the International Federation of Automatic Control*, Seoul, Korea, July 6-11 2008.