ARTICLE IN PRESS

Journal of Systems Architecture xxx (2012) xxx–xxx

Contents lists available at SciVerse ScienceDirect

Journal of Systems Architecture

journal homepage: www.elsevier.com/locate/sysarc

Modeling and efficient solving of extra-functional properties for adaptation in networked embedded real-time systems

Marc Zeller *, Christian Prehofer

Fraunhofer Institute for Communication Systems ESK, Hansastrasse 32, 80686 Munich, Germany

ARTICLE INFO

Article history:
Received 15 February 2012
Received in revised form 9 November 2012
Accepted 10 November 2012
Available online xxxx

Keywords:
Runtime adaptation
Extra-functional properties
Networked embedded systems
Automotive
Constraints

ABSTRACT

In this paper, we focus on modeling and efficient solving of extra-functional properties for embedded systems, in particular automotive systems. We introduce an integrated model of system constraints for efficient computation of software components being allocated to hardware platforms (ECUs), which is a prerequisite for runtime adaptation. For a set of over 126,000 constraints in a realistic automotive system, we compare SAT-solving and different heuristic search algorithms. We show that SAT-solving provides solutions in several seconds, and SAT-solving is more efficient for larger systems, whereas other heuristic search algorithms are slightly better for smaller problems.

1. Introduction

In this paper, we focus on the modeling and efficient solving of extra-functional properties for embedded systems. Our main motivation stems from self-adaptive systems, which can reconfigure their software configurations at runtime [1–3]. Applying these techniques to networked embedded systems poses several new problems due to the extra-functional requirements of embedded systems [4], e.g. resource constraints of the hardware platforms and networks. In particular, we focus on automotive embedded systems, where the main constraints are.

- limited memory resources
- task schedulability
- timing dependencies between software components
- heterogeneous hardware platforms
- different sub-networks connected by a gateway

The goal of this paper is to find a practical model of the constraints above which enables the computation of sound solutions in reasonable time.

These requirements are well studied in embedded systems research and there exist specific and sophisticated mathematical models of the constraints (e.g. [5,6]). Typically, the main goal is to optimize resource usage and to find optimal solutions. Some approaches also consider constraints which cover design time (e.g. wiring constraints) and runtime (e.g. [7,8]). In turn, most of these individual formalisms are computationally highly demanding and do not scale well.

Here, our goal is to find valid solutions efficiently, considering all of the above constraints. We focus on the efficiency and scalability of constraint solving and consider this more important than optimal solutions. Observe that we have to consider all the above constraints at the same time. The motivation for this comes from the fact that adaptation of an embedded system may have to be done during runtime. For instance, in case of failures during runtime, finding a correct solution quickly and within a known time period is more important than unrestricted search for optimal solutions. Even though our main motivation is adaptive systems, the techniques can also be applied in the normal development process as fast solutions where immediate feedback to the developer is important.

Automobiles are a prominent example for a complex networked embedded system. Modern automobiles consist of an increasing number of interconnected electronic devices – so-called Electronic Control Units (ECUs). Innovations within the automotive domain are mainly introduced by software, e.g. driver assistance features. Thus, today we find about 2000 software functions distributed over up to 100 ECUs connected via multiple networks in modern vehicles [9]. Fig. 1 shows the in-vehicle network of a typical upper class automobile.

Enhancing nowadays automotive embedded systems with so-called *Self-* properties* like self-configuration, self-healing, self-optimization or self-protection [10] provides a promising approach for improving the scalability, robustness and flexibility [11]. A

* Corresponding author. Tel.: +49 89 547088 324; fax: +49 89 547088 220.
E-mail address: marc.zeller@esk.fraunhofer.de (M. Zeller).

1383-7621/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.sysarc.2012.11.003

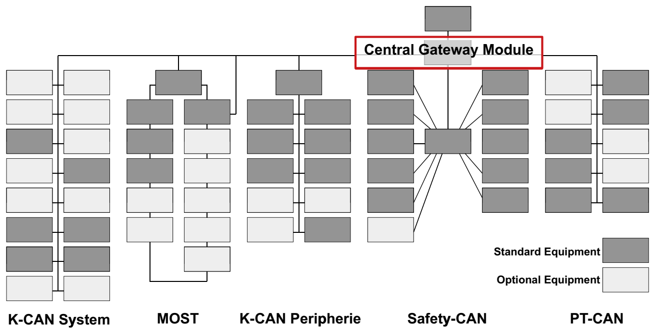

Fig. 1. In-vehicle network topology of a BMW 7-series. (Source: BMW AG, 2005)

prerequisite for such approaches in complex networked embedded systems is the runtime reconfiguration of software component allocation. Since not all possible situations which lead to a reconfiguration of the system can be foreseen during design, the adaptation of the system may have to be calculated during runtime by solving the previously mentioned constraints. Otherwise, systems need to use very conservative fall-back solutions which often mean that all affected features have to be disabled.

We introduce an integrated model of system constraints for quick computation of software component allocation, focusing on automotive embedded systems. Even though the model is simpler than others which focus on specific aspects, its application to realistic automotive system settings leads to more than 2 million variables and more than 126 thousand equations. Secondly, we show that such systems can be solved efficiently in a few seconds on current PC-like hardware. For this, we have compared optimized techniques based on solving the allocation problem formulated as a *SATisfiability (SAT) problem* and different heuristic optimization algorithms, where SAT solving scales better for our set of equations.

The remainder of this paper is organized as follows: first, we will present a formal system model for automotive embedded systems. Section 3 will give a brief definition of terms and explain the principles of runtime adaptation in automotive embedded systems. In Section 4 we present our set of system constraints to define valid allocations in self-adaptive automotive embedded systems. Section 5 gives an overview of methods to solve this set of constraints. Afterwards, we illustrate the benefits of our approach in an experimental evaluation in Section 6. In Section 7 related work is discussed. Finally, the paper is concluded in Section 8.

2. Formal system model

In the following, we introduce a formal definition of automotive embedded systems. An automotive embedded system $\mathcal{A}$ is a heterogeneous distributed real-time system consisting of a set of inputs I (set of sensors), a set of functionalities $F = \{f_1, \ldots, f_n\}$ (*features*), and a set of outputs O (set of actuators). The set of features is realized by a set of *software functions* $SW = \{s_1, \ldots, s_n\}$, where each feature f_i is implemented by a set of software functions SW_{f_i}, which is a sub-set of SW ($SW_{f_i} \subset SW$). Software functions, sensors and actuators are connected to each other in a specified way. This so-called *function network* can be represented by a directed graph $G_f(V_f, E_f)$. The vertices V_f represent software functions, actuators and sensors. The directed edges E_f indicate a data flow from one vertex to another by sending messages. Each vertex may have multiple incoming edges and multiple outgoing edges.

The physical resources of the automotive in-vehicle network are modeled by an undirected graph $G_r(V_r, E_r)$. The vertices V_r represent the vehicles' ECUs ($P = \{p_1, \ldots, p_m\}$) as well as the sensors and actuators. The edges E_r correspond to available communication links (network buses).

The so-called *system configuration c* describes the allocation of the software functions $(s_1, \ldots, s_n)$ to the systems' ECUs $(\{p_1, \ldots, p_m\})$:

$$c : SW \rightarrow P = \{0, 1\}^{n \times m} \tag{1}$$

with $n \cdot m$ variables $x_{i,j}$. If the software function i is assigned to ECU j, then $x_{i,j} = 1$, else it is 0. The general problem of finding a mapping for a set of tasks on a set of resources (often called *knapsack* or *bin-packing problem*) is known to be $\mathcal{NP} - complete$ [12].

Moreover, $\mathcal{A}$ consists of a set of linear constraints $\Psi = \{\psi_1, \ldots, \psi_k\}$ which enable the definition of valid allocations. A linear constraint $\psi \in \Psi$ is defined as a Boolean formula of the following form:

$$\psi = \left(\sum_{i=1}^{n} \sum_{j=1}^{m} a_{i,j} x_{i,j} \right) \circ b \rightarrow \{true, false\} \tag{2}$$

with the Boolean literals $x_{i,j} \in \{0, 1\}$ which represent the allocation of the software function s_i to the ECU p_j, the coefficients $a_{i,j}, b \in \mathbb{R}$, and the operator $\circ \in \{<, \leqslant, =, \geqslant, >\}$.

With this formal system model it is possible to describe the runtime adaptation of a self-adaptive automotive embedded system.

3. System constraints and runtime adaptation

Since automotive embedded systems provide safety-relevant applications, these systems must meet the predefined requirements to guarantee the proper system behavior at all time. Thus, the system configuration c of an automotive system is called *valid* at a certain point of time, if all constraints Ψ are satisfied:

$$\bigwedge_{\psi \in \Psi} \psi = \text{ true } \iff c \text{ is valid} \tag{3}$$

Self-adaptive automotive software systems can be realized by adapting the structure of the system at runtime in response to changes in the environment or within the system itself (*Structural Adaptation*) [13]. In the context of automotive embedded systems, structural adaptation means to find a new mapping of the function network G_f to the existing physical in-vehicle network G_r. To adapt an automotive system in a flexible and adequate way, new system configurations are determined during runtime in order to consider the current system conditions. The aim of the runtime adaptation is to provide a valid system configuration at all times. During runtime, each constraint of the system can be monitored by a control instance. If one of the defined constraints is not met anymore, the structure of the system can be adapted in order to meet the constraints.

Describing a self-adapting system by a set of constraints allows the structural adaptation of the system only within predefined boundaries [4]. Thereby, the system is prevented from emerging invalid allocations and uncontrolled behavior. Since only a small part of all possible system configurations are valid, it is difficult to determine a new, valid system configuration during runtime – especially for an embedded system with restricted resources. Therefore, it is important to find valid allocations (configurations) in reasonable time.

In the next section, we present an integrated set of constraints to define valid allocations in self-adaptive automotive embedded systems which enables the efficient solution of the set of equations with respect to performance.

Please cite this article in press as: M. Zeller, C. Prehofer, Modeling and efficient solving of extra-functional properties for adaptation in networked embedded real-time systems, J. Syst. Architect. (2012), http://dx.doi.org/10.1016/j.sysarc.2012.11.003