Proceedings of the World Congress on Engineering and Computer Science 2012 Vol I
WCECS 2012, October 24-26, 2012, San Francisco, USA

A Column Generation Approach to Scheduling of Real-Time Networks

Ernst Althaus, Sebastian Hoffmann, Joschka Kupilas, and Eike Thaden

***Abstract*—We present an algorithm based on column generation for the real-time scheduling problem of allocating periodic tasks to electronic control units in multiple subsystems connected by a global bus.**

The allocation has to ensure that tasks can be scheduled, and messages between tasks in different subsystems can be transmitted over the global bus and meet their deadlines. Also tasks and messages occurring in a task chain must be scheduled in a way such that the sequence of execution meets their end-to-end deadline.

We show that our approach computes the optimal allocation in our model and due to the column generation approach early provides lower bounds on the optimal value.

***Index Terms*—scheduling, real-time-networks, column-generation, task-chain, end-to-end-deadline.**

I. Introduction

In this paper we look at the scheduling problem arising in the field of manufacturing embedded systems. We are given a hardware architecture with a global bus system, e.g. a FlexRay bus, connecting several processing subsystems $S = \{s_1, s_2, \ldots, s_k\}$. Each electronic control unit (ECU) in a subsystem can be of a specific ECU type $\{et_1, et_2, \ldots, et_\ell\}$, see Figure 1.

Besides a set $T = \{t_1, t_2, \ldots, t_n\}$ of tasks specified by their worst-case execution time (WCET), deadline, period, and memory consumption, where the WCET and memory consumption depend on the ECU type, we are given a set $M = \{m_1, m_2, \ldots, m_v\}$ of messages specified by a single source task, a set of destination tasks, a transmission time, and a deadline, see Figure 4. The scheduling problem arises in allocating every task to a specific ECU in a subsystem such that every message regarding this task can be transmitted over the global bus and both tasks and messages meet their deadlines in every periodic cycle.

We assume that tasks and messages arrive with a fixed rate given as their period, which is greater than or equal to their deadline – the period of a message is determined by the source task – and that we are given deadline monotonic priorities, which was proven to be optimal in our setting [1]. Each task can only be assigned to one ECU and each ECU can execute exactly one task at a time by using preemptive fixed-priority scheduling. Furthermore, the WCET and memory consumption of a task depend on the chosen ECU type of the ECU the task is allocated to. We call a schedule feasible if every task is entirely executed before its deadline.

We present two sufficient models to determine the feasibility of a schedule: first, the computation of the first idle time per ECU by Lehoczky et al. [2] and second, by the computation of the worst-case response time (WCRT) by the well-known fix-point equation by Joseph and Pandya [3].

There are three common bus systems that can be considered for the communication: a token area network (TAN) bus, a controller area network (CAN) bus or a FlexRay bus. We assume that the message communication in every ECU and subsystem is guaranteed and specific gateway ECUs are not necessary. We model a global FlexRay bus so that a message has to be sent over the FlexRay if at least one of the destination tasks is allocated to a different subsystem than the source task.

The problem at hand is of major importance in several industrial sectors, e.g. in aerospace, automotive, and automation industries, as it can save a lot of costs by optimally allocating tasks that have to be definitely executed.

In this paper we formulate the problem as an integer linear program (ILP) and solve it by a column generation approach within a branch and bound framework. We extend the approach of Althaus et al. [4] by integrating the subsystems to the formulation, accelerating the computation if no response times are necessary, handling task chains with their end-to-end deadlines and modeling a global FlexRay bus.

II. Previous Work

The complete design flow from specification models to their distributed execution on hierarchical architectures with different approaches of pre-allocating tasks is described in Bücker et al. [5] and Clark et al. [6] whose repetitive two-tier approach first heuristically distributes the tasks to subsystems, and second, exactly solves the scheduling problem in every subsystem.

Eisenbrand et al. [7] formulated the problem of scheduling pre-allocated tasks in a subsystem as an integer linear program (ILP) which is solved by a standard ILP solver. Althaus et al. [4] improved upon their work by performing a Dantzig–Wolfe decomposition of the ILP formulation by introducing a column generation approach and obtained better running times on large instances.

Manuscript received July 18, 2012; revised August 9, 2012.

This work was partly supported by the German Research Council (DFG) as part of the Transregional Collaborative Research Center "Automatic Verification and Analysis of Complex Systems" (SFB/TR 14 AVACS, www.avacs.org).

Ernst Althaus is with the Max-Planck-Institut für Informatik, Saarbrücken, Germany and the Institut für Informatik, Johannes Gutenberg-Universität, Mainz, Germany (e-mail: ernst.althaus@mpi-inf.mpg.de; ernst.althaus@uni-mainz.de).

Sebastian Hoffmann is with the Institut für Informatik, Johannes Gutenberg-Universität, Mainz, Germany (e-mail: sebastian.hoffmann@uni-mainz.de).

Joschka Kupilas is with the Max-Planck-Institut für Informatik, Saarbrücken, Germany (e-mail: jkupilas@mpi-inf.mpg.de).

Eike Thaden is with the Department für Informatik, Carl von Ossietzky-Universität, Oldenburg, Germany (e-mail: eike.thaden@informatik.uni-oldenburg.de).

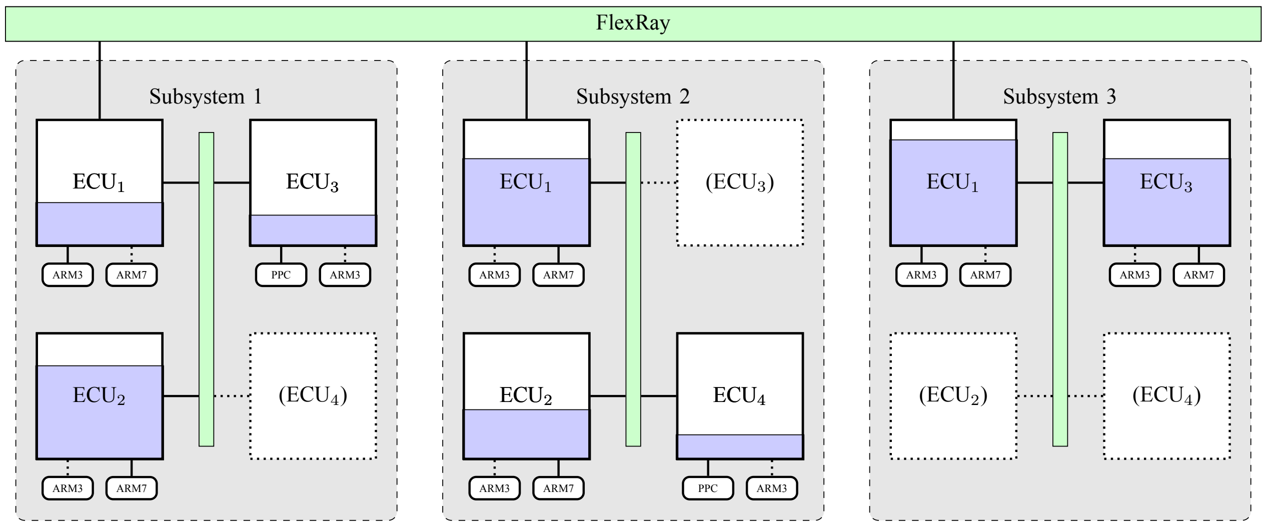


Fig. 1. A hardware network with three subsystems connected by a global FlexRay bus via the specif c gateway ECUs in each subsystem. Upgrading the ECUs increases the cost that has to be minimized.

III. A Column Generation Approach for the Scheduling Model

In this section we describe our column generation approach for solving the previously def ned scheduling problem by presenting an ILP formulation following Althaus et al. [4].

Given a set $S = \{s_1, s_2, \dots, s_k\}$ of subsystems, we call a subset $p \subseteq T$ of tasks a *task pattern* for subsystem $s \in S$ if all tasks in p can be scheduled in the subsystem s and we denote by $P^s \subseteq \mathcal{P}(T)$ the set of all task patterns for s, where $\mathcal{P}$ denotes the power set. Associated with each $p \in P^s$ in the ILP formulation we introduce a binary variable $X_{s,p}$ indicating whether the task pattern p is used for the schedule in subsystem s or not.

The objective function (1) is to minimize the sum of costs $\text{cost}(s,p)$ resulting from the choice of ECU types in each subsystem s needed to execute the allocated task pattern p. We impose two requirements for the task patterns: f rst, exactly one task pattern is used in a subsystem (2), and second, each task has to be assigned to exactly one subsystem (3).

Similarly to the set of task patterns P^s we def ne the set of *message patterns* $Q \subseteq \mathcal{P}(M)$ where $q \in Q$ if all messages in q can be transmitted over the global bus with respect to some bus protocol. We likewise introduce a binary decision variable Z_q for every $q \in Q$ with the requirement that exactly one message pattern must be chosen (6).

Furthermore, we introduce binary variables Y_m for every message $m \in M$ indicating whether the message m is sent over the global bus. Therefore, Y_m is 0 if and only if all destination tasks $m_B \subseteq T$ of message m are in the same subsystem as the source task $m_a \in T$ (4). We also have to ensure that the chosen message pattern contains all messages that have to be transmitted over the bus (5).

Thus, our scheduling problem can be formulated as the following ILP:

$$\min \quad \sum_{s\in S}\sum_{p\in P^s} \text{cost}(s,p)\cdot X_{s,p} \tag{1}$$

$$\text{s.t.} \quad \sum_{p\in P^s} X_{s,p} = 1 \qquad \forall\, s\in S \tag{2}$$

$$\sum_{s\in S}\ \sum_{p\in P^s\,:\,t\in p} X_{s,p} = 1 \qquad \forall\, t\in T \tag{3}$$

$$-Y_m + \sum_{s\in S}\ \sum_{\substack{p\in P^s:\\ m_a\in p\ \wedge\ m_B\setminus p\neq\emptyset}} X_{s,p} = 0 \qquad \forall\, m\in M \tag{4}$$

$$-Y_m + \sum_{q\in Q\,:\,m\in q} Z_q \geq 0 \qquad \forall\, m\in M \tag{5}$$

$$\sum_{q\in Q} Z_q = 1 \tag{6}$$

$$X_{s,p}\in\{0,1\} \qquad \forall\, s\in S, p\in P^s \tag{7}$$

$$Z_q\in\{0,1\} \qquad \forall\, q\in Q \tag{8}$$

$$Y_m\in\{0,1\} \qquad \forall\, m\in M \tag{9}$$

Applying a branch and bound approach, we get bounds from solving the LP relaxations of the ILPs occurring in the branching process. Since the pattern variables arise in exponential number, we use a column generation approach to solve the LP relaxations, see Figure 2.

The remainder of Section III is organized as follows. In Section III-A we expand on the LP relaxation. In Section III-B we explain the branch and bound algorithm. In Section III-C we present the models to scheduling periodic real-time tasks. In Section III-D we show how to handle task chains with end-to-end deadlines. In Section III-E we expand on the FlexRay bus for the global communication.

A. Solving the Relaxation

We relax the integrality constraints (7) of the task patterns $X_{s,p} \in \mathbb{R}$ and add the lower bounds $X_{s,p} \geq 0$ since the upper bounds $X_{s,p} \leq 1$ are implied by the constraints (2).

To state the so-called *master problem*, we make the problem artif cially feasible by introducing a *super subsystem* $\tilde{s} \notin S$ in which tasks are only executable altogether, i.e.

$P^{\tilde{s}} = \{T\}$. The feasibility of the original problem can be decided by inspecting the objective value or the new pattern variable $X_{\tilde{s},T}$, because we assign a cost higher than every optimal solution to it. To satisfy the message pattern constraint (6) we introduce an arbitrary message pattern, e.g. the empty message pattern $Z_\emptyset \in Q$, where no message has to be sent over the global bus. The constructed master problem is feasible by default with

$$Z_\emptyset = X_{\tilde{s},T} = X_{s,\emptyset} = 1 \quad \text{and} \quad Y_m = 0$$

for all $s \in S$ and $m \in M$ with the objective value $\operatorname{cost}(\tilde{s}, T)$.

The master problem is solved with a state-of-the-art simplex method and the so-called *pricing problems* are solved to check if non-basic variables $X_{s,p}$ or Z_q with negative reduced cost can be produced. In this case, the solution of the master problem is not optimal for the original problem – apart from degeneracies – and we have to add the variable to the master problem by generating a new column. This step is repeated until there are no more non-basic variables with negative reduced cost and the last solution to the master problem is also optimal for the original problem, see Figure 2.

The structure of the problem allows us to partition the search for new variables into several independent pricing problems, namely into the *task pricing problems*, the search for a variable $X_{s,p}$ with the most negative reduced cost for a given subsystem s (10), and the *message pricing problem*, the search for a variable Z_q with the most negative reduced cost (15).

First, we generate the task pricing problem for a given subsystem s. By writing d_s for $s \in S$ for the dual variables corresponding to the subsystem constraints (2), d_t for $t \in T$ corresponding to the task constraints (3), and d_m for $m \in M$ corresponding to the message constraints (4) the pricing problem for a given $s \in S$ reads as follows:

$$\begin{aligned}
\text{min} \quad & \operatorname{cost}(s,p) - d_s - \sum_{t \in T} d_t \cdot p_t - \sum_{m \in M} d_m \cdot y_m && (10)\\
\text{s.t.} \quad & y_m - p_{m_a} + p_b \geq 0 \qquad \forall\, m \in M, b \in m_B && (11)\\
& p = \big[p_t\big]_{t \in T} \in P^s && (12)\\
& p_t \in \{0,1\} \qquad \forall\, t \in T && (13)\\
& y_m \in \{0,1\} \qquad \forall\, m \in M && (14)
\end{aligned}$$

with binary variables p_t indicating whether task t is represented in the task pattern p, and y_m indicating whether message m has to be sent over the global bus. Notice that the constraints (11) for a given message $m = (m_a, m_B) \in M$ force y_m to be 1, if the source task m_a is assigned to this subsystem and one of the destination tasks $b \in m_B$ is not.

Similarly, we generate the pricing problem for the message pattern by writing d'_m for the dual variable corresponding to the message constraints (5), and d_Q corresponding to the message pattern constraint (6) to obtain

$$\begin{aligned}
\text{min} \quad & -\sum_{m \in M} d'_m \cdot q_m - d_Q && (15)\\
\text{s.t.} \quad & q = \big[q_m\big]_{m \in M} \in Q && (16)\\
& q_m \in \{0,1\} \qquad \forall\, m \in M && (17)
\end{aligned}$$

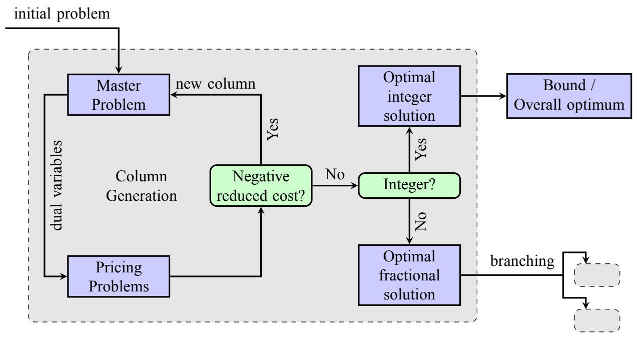


Fig. 2. Our column generation approach inside a branch and bound framework. Computed optimal integer solutions can be used as bounds in the master problems to prune the search tree and reduce the running time.

with binary variables q_m indicating whether message m is transmitted over the global bus.

The constraint (12) has to express the schedulability of the tasks in a subsystem, see Section III-C, as well as the constraint (16) has to ensure the global message transmission due to the different bus types, see Section III-E.

B. Branch and Bound

By solving the LP relaxation we obtain a lower bound on the optimal objective value of the ILP. If the cost of this solution is not smaller than the best integral solution found so far, we can stop. Otherwise, we branch by identifying a fractional assigned task to more than one subsystem and generate two branches: f rst, we predeploy this task to the specif c subsystem, and second, we forbid this task to run in this subsystem. The process sequence of the branches is given by a priority queue in the order of the smallest lower bound f rst.

We achieve a big improvement in the runtime if we reuse columns as starting solutions that do not contradict the branching rule. To interrupt the process of solving pricing problems early we use the Lagrangian bound of Althaus et al. [4] in case that the current objective value cannot be further improved. Additionally, we trade on the integrality of the objective function by rounding up the lower bounds.

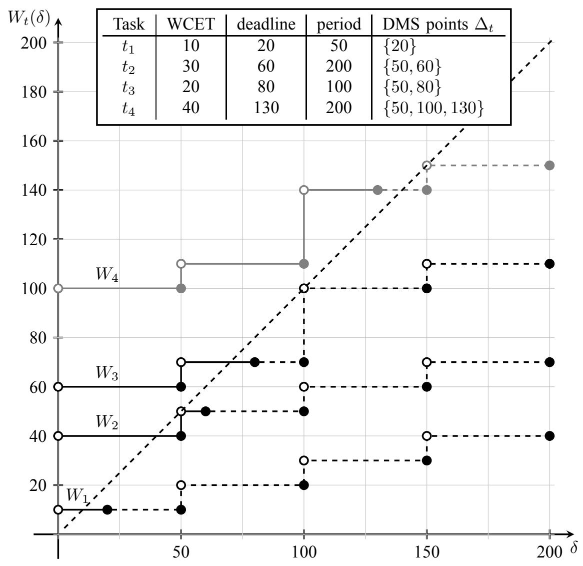
C. Scheduling Periodic Tasks

We are given tasks $t \in T$ with WCET c_t, deadline d_t, and period π_t satisfying $c_t \leq d_t \leq \pi_t$. Eisenbrand et al. [7] state a necessary and suff cient schedulability test by computing the exact WCRTs of the tasks with an ILP formulation and bound them by the specif c deadlines due to the well-known recursive equation from Joseph and Pandya [3].

In the special case of implicit deadlines, i.e. $d_t = \pi_t$, the DMS policy equals the rate monotonic scheduling (RMS) policy which is therefore also optimal under all FPS policies.

In the given case of non-implicit deadlines, i.e. $d_t \leq \pi_t$, the deadline monotonic scheduling (DMS) policy, i.e. the priority is inversely proportional to its deadline, is proven to be optimal under all f xed priority scheduling (FPS) policies, see Burns and Wellings [8].

A suff cient but not necessary schedulability test is provided by Audsley [9] which becomes suff cient and necessary if and only if the considered WCRTs are exact. The workload analysis by Lehoczky et al. [2] presents a pseudo-polynomial,



Task	WCET	deadline	period	DMS points Δ_t
t_1	10	20	50	$\{20\}$
t_2	30	60	200	$\{50, 60\}$
t_3	20	80	100	$\{50, 80\}$
t_4	40	130	200	$\{50, 100, 130\}$

Fig. 3. Cumulative workload demand function $W_t(\delta)$ for some tasks in the interval $[0, \delta]$. It is obvious that (19) has to be checked only in the deadline monotonic scheduling points Δ_t for a schedulability analysis to see that task t_4 cannot be added if the other three tasks are assigned.

necessary and sufficient scheduling test without computing the WCRTs which can also be stated as linear constraints.

The fact that the WCET and memory consumption depend on the chosen ECU type of the ECU the task is assigned to leads to high complexity for the occurring ILPs. To present the concept of Lehoczky et al. we ignore the latter dependencies and assume that there is only one ECU of a fixed ECU type.

For a given task $t \in T$ we compute the *cumulative workload demands* on the ECU over an interval $[0, \delta]$, where 0 is a critical instant for the set $\{t' \mid t' \in \text{hep}(t)\}$, and where $\text{hep}(t)$ denotes all tasks with priority higher than or equal to t, i.e. $d_{t'} \leq d_t$, by

$$W_t(\delta) = \sum_{t' \in \text{hep}(t)} c_{t'} \cdot \left\lceil \frac{\delta}{\pi_{t'}} \right\rceil . \tag{18}$$

Lehoczky et al. [2] show that the entire task set can be scheduled if

$$\max_{t \in T} \min_{\delta \in \Delta_t} \sum_{t' \in \text{hep}(t)} \frac{c_{t'}}{\delta} \cdot \left\lceil \frac{\delta}{\pi_{t'}} \right\rceil \leq 1 \tag{19}$$

with the so-called *deadline monotonic scheduling points*

$$\Delta_t = \{d_t\} \cup \{k \cdot \pi_{t'} \leq d_t \mid t' \in \text{hep}(t), k \in \mathbb{N}\} \tag{20}$$

for task t, see Figure 3. To replace (12) in the task pricing problem we express (19) by the following constraints (21) and (22):

$$-M_t \cdot \alpha_{\delta,t} + \sum_{t' \in \text{hep}(t)} \frac{c_{t'}}{\delta} \cdot \left\lceil \frac{\delta}{\pi_{t'}} \right\rceil \cdot p_{t'} \leq 1 \tag{21}$$

with binary variables $\alpha_{\delta,t} \in \{0, 1\}$ for all $t \in T$ and $\delta \in \Delta_t$ with M_t sufficiently big. The constraints

$$\sum_{\delta \in \Delta_t} \alpha_{\delta,t} \leq |\Delta_t| - 1 \qquad \forall\, t \in T \tag{22}$$

ensure that there has to be at least one point $\delta \in \Delta_t$ where it is not necessary to subtract M_t from the demand to satisfy (19), resp. (21).

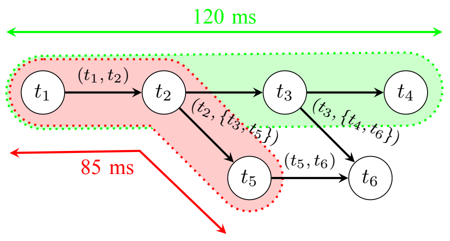


Fig. 4. A network with two simple linear task chains and their end-to-end deadlines.

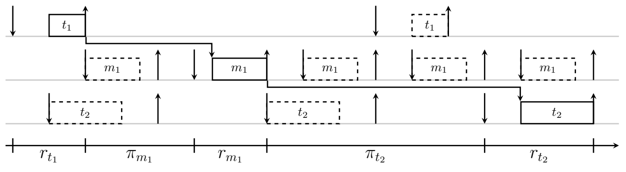


Fig. 5. An example of the worst-case end-to-end latency occurring in the transmission of a message $m_1 = (t_1, t_2)$ over the global bus. Task t_1 misses the first instance of message m_1, and the second instance of m_1 finishes shortly after the start of task t_2. The down arrows indicate the activation times, the up arrows the WCRTs of tasks and messages.

D. End-to-end Deadlines for Task Chains

A task chain $c \subseteq T$ is a set of related tasks with their corresponding messages that has to be executed within a time restriction of an end-to-end deadline. We restrict to the case of simple linear task chains, i.e. $|m_B \cap c| \leq 1$ for every message m with $m_a \in c$, which can always be obtained from more complex ones, see Figure 4. In addition, we assume that every message between tasks in a task chain has to be considered in the end-to-end deadline.

Since tasks of a chain can be distributed over several subsystems, the upcoming messages have to be transmitted over the global bus and therefore the computation of the worst-case end-to-end latency has to take into account the WCRT of all involved tasks and global messages in the chain as well as the periods of the source and destination tasks of the messages. It is necessary to include the periods of the corresponding global messages π_m, i.e. the periods of their source tasks, and their destination tasks, because in the worst case the source task misses the first instance of a message transmission, and the second instance arrives immediately after the start of a destination task, thus will not be read until the next instance of this task, as depicted in Figure 5.

According to the model of Zhu et al. [10] we expand the master problem (1) by

$$\sum_{s \in S} \sum_{\substack{p \in P^s: \\ t \in p \,\cap\, c}} r_{t,p} \cdot X_{s,p} + \sum_{\substack{m \in q: \\ m_a \in c \,\wedge \\ m_B \in c}} \pi_m \cdot Y_m$$
$$+ \sum_{q \in Q} \sum_{\substack{m \in q: \\ m_a \in c \,\wedge \\ m_B \in c}} (r_{m,q} + \pi_{m_B}) \cdot Z_q \leq d_c \tag{23}$$

for all task chains $c \in C$, where C denotes the set of all task chains, and their corresponding end-to-end deadlines d_c.

In order to manage task chains the presented way we need to compute the exact WCRTs of the tasks and messages, and therefore we have to use the scheduling approach of Eisenbrand et al. and cannot use the DMS scheduling approach of Lehoczky et al. presented in Section III-C.

TABLE I
RESULTS OF THE DIFFERENT APPROACHES.

	without task chains		with task chains	
scheduling model	f rst LB	best LB	f rst LB	best LB
memory	24 1.0s	30 3.4s	28 5.3s	30 29.6s
+ slot control	24 0.6s	30 2.6s	28 4.8s	30 24.1s
+ FlexRay bus	24 0.9s	30 3.1s	28 4.9s	30 24.2s
heuristic	30 0.2s		30 0.2s	
optimal value	30		30	

E. The Global FlexRay Bus

There are three common bus systems that can be considered for the global communication: f rst, a token area network (TAN) bus, in which a token is given to the subsystems in a round-robin fashion and the gateway ECU holding the token can send over the bus as discussed by Althaus et al. [4]. Second, a controller area network (CAN) bus, in which the messages gain a priority for a non-preemptive transmission by Davis et al. [11]. Third, a FlexRay bus consisting of a deterministic static segment resembling a time-division multiplexing access (TDMA) fashion, which is analyzed by Lukasiewycz [12], and a CAN bus-like dynamic segment, which we neglect as it is non-deterministic.

In a f rst underapproximation model we only ensure that the number of static slots is suff cient for every message transmitted unaccompanied by extension of the message pricing problem (15). For the FlexRay bus we highly abstract and assume that every slot is suff ciently dimensioned, and that multiplexing is not used. Then we compute a lower bound on the signal's response time r_m in a straightforward manner by bounding it with the maximum time distance between allocated slots for this message.

The local transmission within a subsystem and within the ECUs is neglected in both implemented approaches.

IV. PROOF-OF-CONCEPT EXPERIMENT

A. Setting

As an example, we use a synthetic architecture of two subsystems with at most three ECUs of two different ECU types of cost 10 and 40 in each subsystem, and two copies of the task network of Figure 3 with the given WCETs on the ECU type of cost 10 and the halved WCETs on the ECU type of cost 40. Additionally, we created three signals and one taskchain for a global FlexRay bus of two slots, where one slot is already occupied to transmit one signal from two predeployed tasks to different subsystems.

The optimal allocation in every case uses three ECUs of the cheap ECU type with total cost of 30. More realistic constraints, e.g. that a group of tasks has to be executed on the same ECU, are not activated although they are already implemented.

All experiments were run on one core of a server with two six-core processors (Intel® Core™ i7–970 at 3,2 GHz) with 12 GB RAM. For solving the generated LPs and ILPs we use the commercial LP solver Gurobi Optimizer 4.6.0 [13].

In Table I we show the values and running times of the f rst and best lower bounds obtained by different approaches. For the example without a task chain we used our LP formulation of the approach of Lehoczky et al. [2]. If we consider task chains with end-to-end deadlines, we have to compute the exact response times, and thus use a reimplementation of the approach of Eisenbrand et al. [7]. The heuristic reference value is provided by the approach of Thaden et al. [14].

The approach is able to handle different degrees of underapproximation to fast provide lower bounds. In the memory case we only ensure the schedulability of the periodic tasks and their memory consumption. For the global message transmission we f rst give the results for the described simple slot control, then our abstraction of the FlexRay bus.

B. Evaluation

As expected, the quality of the lower bounds increases with the degree of the model and the runtime. The fact that the weaker model is solved slower will be compensable for larger instances as the ILPs will get larger. In this case a good lower bound is already expectable with a weaker model in less time.

The exact computation of the response times is very time consuming in the case of task chains, because of the more complex ILP formulation in the pricing problems.

A more detailed analysis with more examples is necessary to determine the limits and scalability of the approach.

V. CONCLUSION

We presented a column generation approach for scheduling of real-time networks which satisf es constraints concerning the subsystems, tasks, ECU types, ECUs and messages as well as task chains. This approach extends the approach of Althaus et al. [4] in the sense of a wider perspective of the real-time network.

Our approach can be integrated in a hybrid algorithm that smartly f nds heuristic solutions while proving their quality, and in the best case their optimality.

For solving larger task networks more eff ciently the number of variables and constraints in the pricing problems has to be reduced. We have in mind solving easier, non-exact relaxations of the pricing problems in a combinatorial way, and verify the results appropriately to improve the search for new variables in the column generation approach, provided that we pinpoint which constraints are responsible for the hardness of this problem.

REFERENCES

[1] J. Y. T. Leung and J. Whitehead, "On the complexity of f xed-priority scheduling of periodic, real-time tasks," *Performance Evaluation*, vol. 2, no. 4, pp. 237–250, 1982.

[2] J. P. Lehoczky, L. Sha, and Y. Ding, "The rate monotonic scheduling algorithm: Exact characterization and average case behavior," in *IEEE Real-Time Systems Symposium*, 1989, pp. 166–171.

[3] M. Joseph and P. K. Pandya, "Finding response times in a real-time system." *The Computer Journal*, vol. 29, pp. 390–395, 1986.

[4] E. Althaus, R. Naujoks, and E. Thaden, "A column generation approach to scheduling of periodic tasks," in *Experimental Algorithms - 10th International Symposium, SEA 2011, Proceedings*, ser. LNCS 6630, vol. 1. Springer Berlin, May 2011, pp. 340–351.

[5] M. Büker, W. Damm, G. Ehmen, A. Metzner, I. Stierand, and E. Thaden, "Automating the design f ow for distributed embedded automotive applications: Keeping your time promises, and optimizing costs, too," in *Proc. International Symposium on Industrial Embedded Systems (SIES'11)*, 2011.

[6] B. Clark, I. Stierand, and E. Thaden, "Cost-minimal pre-allocation of software tasks under real-time constraints," in *Proceedings of the 2011 ACM Symposium on Research in Applied Computation (RACS 2011)*, R. E. Gantenbein and T. W. W. Kuo, Eds., Miami, Florida, 2011, pp. 77–83.

[7] F. Eisenbrand, W. Damm, A. Metzner, G. Shmonin, R. Wilhelm, and S. Winkel, "Mapping task-graphs on distributed ecu networks: Eff cient algorithms for feasibility and optimality," in *Proceedings of the 12th IEEE Conference on Embedded and Real-Time Computing Systems and Applications*. IEEE Computer Society, 2006.

[8] A. Burns and A. Wellings, *Real-time systems and programming languages: Ada 95, real-time Java, and real-time POSIX*, ser. International computer science series. Addison-Wesley, 2001.

[9] N. C. Audsley, "Deadline monotonic scheduling," 1990.

[10] Q. Zhu, Y. Yang, M. D. Natale, E. Scholte, and A. L. Sangiovanni-Vincentelli, "Optimizing the software architecture for extensibility in hard real-time distributed systems," *IEEE Trans. Industrial Informatics*, vol. 6, no. 4, pp. 621–636, 2010.

[11] R. I. Davis, A. Burns, R. J. Bril, and J. J. Lukkien, "Controller area network (CAN) schedulability analysis: Refuted, revisited and revised," *Real-Time Systems*, vol. 35, no. 3, pp. 239–272, 2007.

[12] M. Lukasiewycz, M. Glaß, J. Teich, and P. Milbredt, "FlexRay schedule optimization of the static segment," in *CODES+ISSS*, W. Rosenstiel and K. Wakabayashi, Eds. ACM, 2009, pp. 363–372.

[13] Gurobi Optimization, Inc., "Gurobi optimizer reference manual," 2012. [Online]. Available: http://www.gurobi.com

[14] E. Thaden, H. Lipskoch, A. Metzner, and I. Stierand, "Exploiting gaps in f xed-priority preemptive schedules for task insertion," in *Proceedings of the 16th (IEEE) International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*. (IEEE) Computer Society, 2010, pp. 212–217.