Improvements of a hybrid ILP-CP Benders decomposition for mapping and scheduling task DAGs on heterogeneous architectures

Andreas Emeretlis · George Theodoridis · Panayiotis Alefragis · Nikolaos Voros

Keywords task graph timetabling · Benders decomposition · heterogeneous resources · proven optimal solutions · cut generation

Extended abstract

In this work, an improved method for optimally mapping applications represented as Directed Acyclic Graphs (DAGs), on heterogeneous multi-core platforms is presented. The set of nodes $V = v_1, v_2, ..., v_n$ of the DAG corresponds to the tasks of the application, while edges represent data dependencies between tasks. The target platform consists of $P = 1, 2, ..., m$ cores. The problem can be categorized as a timetable problem as it requires the simultaneous selection of performing resources and the scheduling of the assigned tasks for each resource, while satisfying dependency constraints. We extend the logic-based Benders decomposition approach presented in [1] that combines Integer Linear Programming (ILP) and Constraint Programming (CP) models with new cuts in order to improve the solution quality and/or the algorithm execution time. In the decomposition, the Master Problem (MP) is finding an assignment of

A. Emeretlis
Electrical and Computer Engineering Department, University of Patras, Greece
E-mail: emeretlis@ce.upatras.gr
G. Theodoridis
Electrical and Computer Engineering Department, University of Patras, Greece
E-mail: theodor@ce.upatras.gr
P. Alefragis
Computers and Informatics Engineering Department, Technological Educational Institute of Western Greece, Greece
E-mail: alefrag@teiwest.gr
N. Voros
Computers and Informatics Engineering Department, Technological Educational Institute of Western Greece, Greece
E-mail: voros@teiwest.gr

tasks to cores and the sub-problem is solving a Sequencing Problem (SP) per core.

Due to the decomposition, there is a loose interaction between the two submodels. The major drawback of this approach is that the MP assigns to the fastest cores more tasks per moving time window the core can handle as it ignores the sequencing problem. To overcome this inefficiency the MP was enriched with a partial subproblem relaxation that tries to provide guidance in generating more prominent assignment solutions. This was achieved through the preprocessing algorithm of [2], which provides information about the sequential execution of the tasks and its impact on the global solution. This way, many solutions are excluded that will certainly lead to worse makespan values after solving the SP.

In this paper, we introduce extra constraints in the MP, trying to reflect the impact of the sequencing of the direct predecessors and successors of every node of the DAG. Specifically, the constraints state that the start time of a task is derived not only by the precedence constraints but also by the sum of the processing time of its direct predecessors that are assigned on the same core. Equivalently, a task should complete its execution early enough so that if its direct successors are assigned on the same core, their aggregated execution time should not exceed their deadlines. We introduce two new sets of constraints that are a relaxation of the above statements. More spesifically, the constraints do not take into account the actual start time of the predecessors, which is a decision variable whose value is known only after the solution of the MP. Instead, they use a relaxed value of the start time, which is the release time (RL) of the task.

This value is easily computed in advance by traversing the graph. The same holds for the direct successors of a task, where the deadline (DL) is considered, which is also derived from the graph. More formally

$$\forall v_i \in V, \forall p \in P \qquad \min_{j \in pred(i)} RL_j + \sum_{j \in pred(i)} D_{jp} x_{jp} \leq t_i^s \tag{1}$$

$$\forall v_i \in V, \forall p \in P \;\; t_i^s + \sum_{p \in P} D_{ip} x_{ip} \leq \max_{j \in succ(i)} DL_j - \sum_{j \in succ(i)} D_{jp} x_{jp} \tag{2}$$

where *pred* and *succ* are the sets that contain the direct predecessors and successors of a node, respectively. Moreover, x_{ip} is a binary decision variable that equals to 1 when task v_i is assigned on core p, otherwise $x_{ip} = 0$, D_{ip} is the execution time of task v_i in core p, while t_i^s is an integer variable that denotes the start time of the task v_i. The inclusion of the above equations enforces the MP to exploit the potential parallelism by avoiding to assign many tasks on the same core. However, as this is only a relaxation, those constraints do not take into account their impact to the start time of tasks that are not directly connected to the task that they refer to. Therefore, their combination with the already included relaxation of [2] is essential so that inefficient solutions are excluded a priori.

Tasks	Cores	ILP	[1]	*Prop.*
20	2	35.8	16.2	**10.1**
20	5	197.9	32.1	**7.8**
30	4	–	109.9	**66.8**
30	6	391.5	101.7	**36.3**
50	8	910.5	268.6	**44.0**
50	10	1675.6	154.0	**31.4**
Total Mean		263.0	76.2	**31.3**
Solved/Unsolved		17/43	52/8	**54/6**

Table 1 Mean time in sec. and number of proven optimal solutions before 7,200 sec. time limit

We have implemented all the evaluated models using the FICO Xpress Optimization suite [3] and performed extensive tests with both generated and real embedded applications task graphs. Sixty graphs with different characteristics were used, ten per each type. Table 1 presents the mean solution time and the number of achieved proven optimal solutions, compared with a full ILP model and the proposed methods in [1].

Based on the results, it is clear that the proposed constraints, while managing to solve just two more problems compared with the method in [1], it improves the mean execution time by more than 50% managing to be nine times faster than the ILP solver. It is also proved that the introduced relaxation is beneficial for the model even though it slightly increases its complexity. The paper presents work in progress and it is in our plans to extend this work to introduce new constraints and support worst case execution time (WCET) mapping and scheduling.

Acknowledgments

This work is co-funded by the European Union under the Horizon 2020 Framework Program under grant agreement ICT-2015-688131, project "WCET-Aware Parallelization of Model-Based Applications for Heterogeneous Parallel Systems (ARGO)".

References

1. A. Emeretlis, G. Theodoridis, P. Alefragis, N. Voros, A Logic-Based Benders Decomposition Approach for Mapping Applications on Heterogeneous Multicore Platforms, *ACM Trans. Embed. Comput. Syst.* 15, 1, 1-28, (2016)
2. Maravelias Christos T., A decomposition framework for the scheduling of single- and multi-stage processes, *Comput. Chem. Eng.* 30, 3, 407420, (2006)
3. Fair Isaac Corporation FICO. FICO Xpress Optimization Suite. (2013).