Technical Report

Intra-Type Migrative Scheduling of Implicit-Deadline Sporadic Tasks on Two-Type Heterogeneous Multiprocessor

Gurulingesh Raravi

Björn Andersson

Konstantinos Bletsas

HURRAY-TR-110402

Version:

Date: 04-01-2011

Intra-Type Migrative Scheduling of Implicit-Deadline Sporadic Tasks on Two-Type Heterogeneous Multiprocessor

Gurulingesh Raravi, Björn Andersson, Konstantinos Bletsas

IPP-HURRAY!

Polytechnic Institute of Porto (ISEP-IPP)

Rua Dr. António Bernardino de Almeida, 431

4200-072 Porto

Portugal

Tel.: +351.22.8340509, Fax: +351.22.8340509

E-mail:

http://www.hurray.isep.ipp.pt

Abstract

Consider the problem of scheduling a set of implicit-deadline sporadic tasks to meet all deadlines on a two-type heterogeneous multiprocessor platform. Each processor is either of type-1 or type-2 with each task having different execution time on each processor type. Jobs can migrate between processors of same type (referred to as intra-type migration) but cannot migrate between processors of different types. We present a new scheduling algorithm namely, LP-Relax(THR) which offers a guarantee that if a task set can be scheduled to meet deadlines by an optimal task assignment scheme that allows intra-type migration then LP-Relax(THR) meets deadlines as well with intra-type migration if given processors 1/THR as fast (referred to as speed competitive ratio) where THR <= 2/3.

Intra-Type Migrative Scheduling of Implicit-Deadline Sporadic Tasks on Two-Type Heterogeneous Multiprocessor

Gurulingesh Raravi (Speaker)[1], Björn Andersson[21] and Konstantinos Bletsas[1]

1 Introduction

Consider the problem of scheduling a set of implicit-deadline sporadic tasks to meet all deadlines on a two-type heterogeneous multiprocessor platform. Each processor is either of type-1 or type-2 with each task having different execution time on each processor type. Jobs can migrate between processors of same type (referred to as *intra-type migration*) but cannot migrate between processors of different types. We present a new scheduling algorithm namely, LP-Relax(THR) which offers a guarantee that if a task set can be scheduled to meet deadlines by an optimal task assignment scheme that allows intra-type migration then LP-Relax(THR) meets deadlines as well with intra-type migration if given processors $\frac{1}{THR}$ as fast (referred to as *speed competitive ratio*) where $THR \leq \frac{2}{3}$.

2 LP-Relax Algorithm

The system considered in this paper is as follows: (i) **Tasks (denoted as τ):** n implicit-deadline sporadic tasks, i.e., for each task, its deadline = its minimum inter-arrival time (denoted as T_i) (ii) **Utilization:** The utilization of a task $\tau_i \in \tau$ on a processor of type-t (where $t \in \{1, 2\}$) is given by U_i^t where $U_i^t = \frac{C_i^t}{T_i}$ (C_i^t denotes the worst-case execution time of a task τ_i on a processor of type-t) and (iii) **Processors:** The platform consists of m processors of which mP^1 processors are of type-1 and mP^2 processors are of type-2. The following assumptions are made: (i) **Intra-Type Migrative:** The jobs released by tasks can only migrate between processors of the same type but not between processors of different types, (ii) **No job parallelism:** A job can be executing on at most one processor at any given point in time and (iii) **Independent tasks:** The execution of jobs are independent, i.e., they neither share any resources nor have data dependency.

Let THR denote a real number number in the range $(0, 2/3]$, selectable by the algorithm designer. Based on THR, let us define the following disjoint sets:

$$H12 = \{\tau_i \in \tau : U_i^1 > THR \ \wedge \ U_i^2 > THR\} \quad (1)$$

$$H1 = \{\tau_i \in \tau : U_i^1 \leq THR \ \wedge \ U_i^2 > THR\} \quad (2)$$

$$H2 = \{\tau_i \in \tau : U_i^1 > THR \ \wedge \ U_i^2 \leq THR\} \quad (3)$$

$$L = \{\tau_i \in \tau : U_i^1 \leq THR \ \wedge \ U_i^2 \leq THR\} \quad (4)$$

[1] {ghri, baa, ksbs}@isep.ipp.pt. CISTER-ISEP Research Center, Polytechnic Institute of Porto, Rua Dr. António Bernardino de Almeida 431, 4200-072 Porto, Portugal.

[2] baandersson@sei.cmu.edu. Software Engineering Institute, Carnegie Mellon University, 4500 Fifth Avenue Pittsburgh, PA 15213-2612, USA.

function: LP-Relax(THR)
1 Form sets $H12$, $H1$, $H2$, L as defined by Eq. 1-4
2 **if** ($H12 \neq \phi$) **then** declare FAILURE **end if**
3 $UP^1 := UP^2 := 0$ //used capacities of two processor types
4 $\tau p^1 :=$ assign($H1, UP^1, 1$)
5 **if** ($\tau p^1 \neq H1$) **then** declare FAILURE **end if**
6 $\tau p^2 :=$ assign($H2, UP^2, 2$)
7 **if** ($\tau p^2 \neq H2$) **then** declare FAILURE **end if**
8 LPR := formulate_LP($L, UP^1, UP^2, mP^1, mP^2$)
9 (Z, X) = LP-Solver(LPR) //Solve LPR using an LP-Solver
10 **if** ($Z > 1$) **then** declare FAILURE
11 **for each** task $\tau_i \in L$ **do** //$x_{i,1}, x_{i,2} \in X$
12 **if** ($x_{i,1} = 1 \wedge x_{i,2} = 0$) **then**
13 $\tau p^1 := \tau p^1 \cup \{\tau_i\}$ //assign τ_i to processor type-1
14 $UP^1 := UP^1 + U_i^1$
15 **end if**
16 **if** ($x_{i,1} = 0 \wedge x_{i,2} = 1$) **then**
17 $\tau p^2 := \tau p^2 \cup \{\tau_i\}$ //assign τ_i to processor type-2
18 $UP^2 := UP^2 + U_i^2$
19 **end if**
20 **end for**
21 **if** ($\exists i$ such that $0 < x_{i,1} < 1$ and $0 < x_{i,2} < 1$) **then**
22 Let τ_f denote that task and $t1$ denote its favorite and $t2$ its non-favorite type, i.e., $U_i^{t1} \leq U_i^{t2}$ (where $t1 = 1$, $t2 = 2$ or $t1 = 2$, $t2 = 1$)
23 $UP^{t1} := UP^{t1} + x_{f,t1} \cdot U_f^{t1}$
24 $UP^{t2} := UP^{t2} + x_{f,t2} \cdot U_f^{t2}$
25 **if** ($UP^{t1} - x_{f,t1} \cdot U_f^{t1} + U_f^{t1} \leq mP^{t1}$) **then**
26 $UP^{t1} := (UP^{t1} - x_{f,t1} \cdot U_f^{t1} + U_f^{t1})$
27 $\tau p^{t1} := \tau p^{t1} \cup \{\tau_f\}$
28 declare SUCCESS
29 **end if**
30 **if** ($UP^{t2} - x_{f,t2} \cdot U_f^{t2} + U_f^{t2} \leq mP^{t2}$) **then**
31 $UP^{t2} := (UP^{t2} - x_{f,t2} \cdot U_f^{t2} + U_f^{t2})$
32 $\tau p^{t2} := \tau p^{t2} \cup \{\tau_f\}$
33 declare SUCCESS
34 **end if**
35 declare FAILURE
36 **else**
37 declare SUCCESS
38 **end if**

Figure 1: The LP-Relax(THR) algorithm for assigning tasks on to a two-type heterogeneous multiprocessor platform.

function assign(ts: set of tasks; U: out current utilization of processors; k: favorite processor type of ts) return set of tasks
1 assigned_tasks := $\emptyset$
2 Use any order for tasks ts, but maintain it during the execution of the function assign.
3 τ_i := first task in ts
4 **if** ($U + U_i^k \leq mP^k$) **then**
5 $U := U + U_i^k$
6 assigned_tasks := assigned_tasks $\cup \{\tau_i\}$
7 **if** (remaining tasks exist in ts) **then**
8 τ_i := next task in ts
9 **go to** line 4.
10 **end if**
11 **end if**
12 **return** assigned_tasks

(a) Assigning heavy tasks to type-k processors

function formulate_LP(L, UP^1, UP^2) return the formulation
1 Let $myLP$ denote the following linear program
2 Minimize Z subject to:
3 $\forall \tau_i \in L : x_{i,1} + x_{i,2} = 1$
4 $\left(\sum_{\tau_i \in L} x_{i,1} \cdot U_i^1\right) + UP^1 \leq mP^1 \cdot Z$
5 $\left(\sum_{\tau_i \in L} x_{i,2} \cdot U_i^2\right) + UP^2 \leq mP^2 \cdot Z$
6 $\forall \tau_i \in L : x_{i,1}$ and $x_{i,2}$ are non-negative real numbers
7 **return** $myLP$

(b) LP formulation for assigning light tasks to processors

Figure 2: Sub-routines used by LP-Relax(THR) algorithm while assigning the task set.

Note that $H12 \cup H1 \cup H2 \cup L = \tau$.

A task is termed *heavy* on a processor type if its utilization on that processor type is greater than THR. The set $H12$ represents those tasks that are heavy on both types of processors and hence these tasks cannot be assigned to any processor type to meet deadlines if the processor speed would be scaled by THR. $H1$ and $H2$ represent those tasks that are heavy on processors of type-2 and type-1 respectively and hence they must be assigned to processors of type-1 and type-2 respectively. L represents those tasks that are not heavy (termed *light*) on both the processor types and hence they can be assigned to processors of any type.

The new algorithm is shown in Figure 1 which uses sub-routines *assign()* and *formulate_LP()* shown in Figure 2(a) and Figure 2(b) respectively.

The algorithm first assigns tasks that are heavy on a certain processor type say, type-1 to processors of type-2 and vice versa (if the schedulability test permits). We can formulate the problem of assigning *light* tasks to processors as an Integer Linear Program (ILP) and then relax it to Linear Program (LP). For assigning light tasks, LP-Relax solves this relaxed LP formulation using an LP-solver which gives the optimal solution at the *vertex* of the *convex* region. This vertex solution gives the assignment (if the schedulability test permits) of all the light tasks to unique processor type except at most

one task say, τ_{fract}, which may be assigned to both the processor types – see Lemma 6 in [5]. Finally, if τ_{fract} was assigned to two processor types then LP-Relax algorithm (re-)assigns τ_{fract} to a single processor type (if the schedulability test permits). Once the task assignment is done, an optimal scheduling algorithm for identical multiprocessors, such as Pfair [1] can be used on each processor type to schedule the tasks.

We now list some important properties about the performance guarantee offered by LP-Relax algorithm. The proofs of these properties are omitted here due to space constraint and can be found in Section 4 in [5].

Theorem 1 *The speed competitive ratio of LP-Relax(2/3) is at most 1.5.*

The following theorem states the speed competitive ratio of LP-Relax algorithm when the maximum utilization of any task on any type of processor (type-1 or type-2) is upper bounded by α, i.e., $\forall \tau_i \in \tau : U_i^1 \leq \alpha \wedge U_i^2 \leq \alpha$ where $0 < \alpha \leq \frac{2}{3}$.

Theorem 2 *The speed competitive ratio of LP-Relax($\frac{m-\alpha}{m}$) is at most ($\frac{m}{m-\alpha}$).*

Corollary 1 *If a task set τ is schedulable by any algorithm that allows intra-type migration on a computing platform Π of m processors (having mP^1 processors of type-1 and mP^2 processors of type-2) then LP-Relax($\frac{2}{3}$) also schedules τ on a computing platform Π' of at most $m+1$ processors (having either mP^1+1 processors of type-1 and mP^2 processors of type-2 or mP^1 processors of type-1 and mP^2+1 processors of type-2).*

Corollary 2 *If migration of tasks is allowed between processors of different types (i.e., fractional assignment of tasks is allowed – for a task τ_i, $0 < x_{i,1} < 1$ and $0 < x_{i,2} < 1$) then LP-Relax (with THR=1) is optimal.*

Note: Though Corollary 2 is an important result, we would like to mention that it is not the first of its kind. A feasibility condition exists [3] for checking the schedulability of a task set on a heterogeneous multiprocessor platform (having more than two types of processors) that allows task migration between processors of any type.

Acknowledgments

This work was partially funded by the Portuguese Science and Technology Foundation (Fundação para a Ciência e a Tecnologia - FCT), REHEAT and REJOIN projects.

References

[1] J. Anderson and A. Srinivasan, *Pfair Scheduling: Beyond Periodic Task Systems*, 7th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (2000).

[2] S. Baruah, *Task partitioning upon heterogeneous multiprocessor platforms*, 10th IEEE International Real-Time and Embedded Technology and Applications Symposium (2004).

[3] S. Baruah, *Feasibility Analysis of Preemptive Real-Time Systems upon Heterogeneous Multiprocessor Platforms*, 25th IEEE International Real-Time Systems Symposium (2004).

[4] C. N. Potts, *Analysis of a linear programming heuristic for scheduling unrelated parallel machines*, Discrete Applied Mathematics (1985).

[5] G. Raravi, B. Andersson and K. Bletsas, *Scheduling Implicit-Deadline Sporadic Tasks on Two-Type Heterogeneous Multiprocessor Platform with Restricted Migration*, Tech. Report, CISTER/ISEP, Polytechnic Institute of Porto, HURRAY-TR-110109, (2011).