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), Björn Andersson, and Konstantinos Bletsas

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 $\frac{1}{1+THR}$ 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 $\tau$): $n$ 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 $U_t^{\tau_i}$ where $U_t^{\tau_i} = \frac{C_t^{\tau_i}}{T_i}$ ($C_t^{\tau_i}$ denotes the worst-case execution time of a task $\tau_i$ 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 in the range $(0, 2/3]$, selectable by the algorithm designer. Based on THR, let us define the following disjoint sets:

\begin{align*}
H_{12} &= \{\tau_i \in \tau : U_1^{\tau_i} > THR \land U_2^{\tau_i} > THR\} \\
H_1 &= \{\tau_i \in \tau : U_1^{\tau_i} \leq THR \land U_2^{\tau_i} > THR\} \\
H_2 &= \{\tau_i \in \tau : U_1^{\tau_i} > THR \land U_2^{\tau_i} \leq THR\} \\
L &= \{\tau_i \in \tau : U_1^{\tau_i} \leq THR \land U_2^{\tau_i} \leq THR\}
\end{align*}

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 $H_{12}$, $H_1$, $H_2$, L as defined by Eq. 1-4
2 if $(H_{12} \neq \emptyset)$ then declare FAILURE end if
3 $U^P_1 := U^P_2 := 0$ //used capacities of two processor types
4 $\tau^1 := \text{assign}(H_1, U^P_1)$, 1
5 if $(\tau^1 \neq H_1)$ then declare FAILURE end if
6 $\tau^2 := \text{assign}(H_2, U^P_2)$, 2
7 if $(\tau^2 \neq H_2)$ then declare FAILURE end if
8 LP := formulate LP($L$, $U^P_1$, $U^P_2$, $m^P_1$, $m^P_2$)
9 $(Z, X) = LPSolver(LP)$ //Solve LP using an LP-Solver
10 if $(Z > 1)$ then declare FAILURE end if
11 for each task $\tau_i \in L$ do $//x_{i,1}, x_{i,2} \in X$
12 if $(x_{i,1} = 1 \land x_{i,2} = 0)$ then
13 $\tau^1 := \tau^1 \cup \{\tau_i\}$ //assign $\tau_i$ to processor type-1
14 $U^P_1 := U^P_1 + \tau^1$
15 end if
16 if $(x_{i,1} = 0 \land x_{i,2} = 1)$ then
17 $\tau^2 := \tau^2 \cup \{\tau_i\}$ //assign $\tau_i$ to processor type-2
18 $U^P_2 := U^P_2 + \tau^2$
19 end if
20 end for

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)
1 assigned_tasks := \emptyset
2 Use any order for tasks ts, but maintain it during the execution of the function assign.
3 $\tau_i := \text{first task in ts}$
4 if $(U + U^P_1 \leq m^P_k)$ then
5 $U := U + U^P_1$
6 assigned_tasks := assigned_tasks $\cup \{\tau_i\}$
7 if (remaining tasks exist in ts) then
8 $\tau_i := \text{next task in ts}$
9 go to line 4.
10 end if
11 end if
12 return assigned_tasks

(a) Assigning heavy tasks to type-k processors

(b) LP formulation for assigning light tasks to processors

function formulate(LP($L$, $U^P_1$, $U^P_2$))
1 return the formulation
2 Let myLP denote the following linear program
3 Minimize $Z$ subject to:
4 $\forall \tau_i \in L : x_{i,1}, x_{i,2} = 1$
5 $\sum_{\tau_i \in L} x_{i,1} \cdot U^P_1 + U^P_1 \leq m^P_1 \cdot Z$
6 $\sum_{\tau_i \in L} x_{i,2} \cdot U^P_2 + U^P_2 \leq m^P_2 \cdot Z$
7 $\forall \tau_i \in L : x_{i,1}$ and $x_{i,2}$ are non-negative real numbers
8 return myLP

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

Note that $H_{12} \cup H_1 \cup H_2 \cup L = \tau$.

A task is termed heavy on a processor type if its utilization on that processor type is greater than THR. The set $H_{12}$ 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. $H_1$ and $H_2$ 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, $\tau_{\text{fract}}$, which may be assigned to both the processor types – see Lemma 6 in [5]. Finally, if $\tau_{\text{fract}}$ was assigned to two processor types then LP-Relax algorithm (re-)assigns $\tau_{\text{fract}}$ 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 $\alpha$, i.e., $\forall \tau_i \in \tau : U^1_i \leq \alpha \land U^2_i \leq \alpha$ 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 $\tau$ is schedulable by any algorithm that allows intra-type migration on a computing platform $\Pi$ 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 $\tau$ on a computing platform $\Pi'$ 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 $\tau_i$, $0 < x_{i,1} < 1$ 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 (Fundaçao para a Ciencia e a Tecnologia - FCT), REHEAT and REJOIN projects.

**References**


