Approximation algorithm for scheduling a chain of tasks on heterogeneous systems

Massinissa Ait Aba, Lilia Zaourar, Alix Munier

Approximation Algorithm for Scheduling a Chain of Tasks on Heterogeneous Systems

Massinissa Ait Aba[1(✉)], Lilia Zaourar[1], and Alix Munier[2]

[1] CEA, LIST, Computing and Design Environment Laboratory,
91191 Gif sur Yvette Cedex, France
Massinissa.aitaba@cea.fr
[2] LIP6-UPMC, 4 place Jussieu, 75005 Paris, France

Abstract. This paper presents an efficient approximation algorithm to solve the task scheduling problem on heterogeneous platform for the particular case of the linear chain of tasks. The objective is to minimize both the total execution time (makespan) and the total energy consumed by the system. For this purpose, we introduce a constraint on the energy consumption during execution. Our goal is to provides an algorithm with a performance guarantee. Two algorithms have been proposed; the first provides an optimal solution for preemptive scheduling. This solution is then used in the second algorithm to provide an approximate solution for non-preemptive scheduling. Numerical evaluations demonstrate that the proposed algorithm achieves a close-to-optimal performance compared to exact solution obtained by *CPLEX* for small instances. For large instances, *CPLEX* is struggling to provide a feasible solution, whereas our approach takes less than a second to produce a solution for an instance of 10000 tasks.

Keywords: Linear chain of tasks · Makespan · Energy
Approximation algorithm

1 Introduction

Today, our daily life requires massive calculations on different computing systems (desktop, data centers) to perform various needs such as physical simulations or google searches. In order to improve the performance of these systems while keeping their energy consumption reasonable, heterogeneous system has merged. This heterogeneous architecture combines both processing elements (such as CPUs, GPUs), and reconfigurable logic (FPGAs).

However, taking advantage of such heterogeneous systems requires efficient use of resources to make profit from the performance of each part for application execution. Thus efficient scheduling of task's applications is difficult problem often faced by designers and engineers using these complex systems. In fact, with the complexity of applications and architectures, it becomes increasingly difficult

to distribute the tasks application effectively. More than a simple load balancing problem, heterogeneity leads to consider efficient scheduling techniques to take account of the different resources specificities. The objective of this work is to determine an efficient scheduling of a parallel application on a heterogeneous resources system in order to minimize both the total execution time (makespan) and the energy consumption. For this purpose, we introduce a constraint on the total energy consumed by the system. We consider in this work, a chain of tasks and communication delay. We conducted this research using the fully heterogeneous micro-server system Christmann RECS©|BOX [3]. The rest of the paper is organized as follows. Section 2 discusses some previous efforts in scheduling parallel application on heterogeneous systems, with a focus on makespan and energy minimization. Section 3 presents a detailed description of the mathematical model proposed. In Sect. 4, we present an optimal algorithm for a chain of preemptive task. In Sect. 5 we describe the proposed algorithm for non-preemptive scheduling and approximation ratio we obtain. Section 6 shows some preliminary numerical results. The paper ends with a conclusion in Sect. 7.

2 Related Work

Due to its key importance on performance, the task scheduling problem on heterogeneous platform has been extensively studied and numerous methods have been reported in the literature. They proposed various models and techniques such as dynamical voltage scaling (DVS), list algorithms and genetic heuristics to optimize essentially two main objectives: makespan and energy consumption. Xie et al. [12], demonstrate that minimizing schedule length of a DAG-based parallel application with energy consumption constraint on heterogeneous distributed systems is a nondeterministic polynomial-hard optimization problem. They decompose the problem in two sub-problems beginning by treating the problem of the energy constraint. At each task assignment phase, the energy consumption constraint of the application can always be satisfied by supposing that the unassigned tasks are assigned to the processor with the minimum energy consumption. Then, they proceed to the minimization of makespan, assigning tasks to processors using the earliest finish time (EFT).

Authors in [13], considered the objective of maximizing the probability of completing tasks before a deadline D and to satisfy an energy constraint with execution times and stochastic communications delays. Zhang et al. [15] have treated the problem of robustness under energy constraint. The aim is to maximize system reliability by repairing runtime errors caused by various reasons such as hardware flaws and program bugs while maintaining the energy constraint. Authors in [16] began by giving an IP (Integer Programming) formulation of the problem, then a three-phase algorithm is proposed using the Dynamic Power Management (DPM) and DVS techniques. Several heuristics (iterative, Greedy, random, ...) are proposed in [8] for the problem of scheduling on heterogeneous processors that can change their frequencies among a set of possible values. The objective is to minimize the temperature more than performance and energy of

the system. A three-phase list algorithm is proposed by Fard et al. [2]. They began by analyzing and classifying the different objectives and their impacts on the optimization process. The objective is to find a solution that minimizes up to four objectives (energy, makespan, reliability, economic cost).

Many works have also been done using genetic algorithms. Authors in [5], proposed the ECS heuristic (Energy Concious Heuristic) which is used in [7] to form a hybrid approach with the multi-objective genetic algorithm. This approach provides a set of Pareto solutions. More recently in [14], authors proposed a new genetic algorithm to study both objectives at once. Authors in [9–11] also use game theory strategies to prove the existence of Nash equilibrium and find a Pareto point.

However, all the aforementioned works did not consider approximation techniques. To the best of our knowledge, we propose the first algorithm with a guarantee of performance. Our model is inspired by [1], where authors seek to minimize the energy consumed during execution by imposing a Deadline D on completion time. In addition, we consider in this work communication cost between tasks and processing elements. Preliminary results on modeling applications and heterogeneous platforms have been presented in [6], we focus in this work on tasks chain to determine a performance guarantee algorithm.

3 Model

This study considers a fully connected heterogeneous multiprocessor platform in which M is a set of m heterogeneous processing elements (GPU, CPU, FPGA...) noted PE. Each element $PE_k \in M$ is characterized by its execution frequency $f_j \geqslant 1$, $j = \overline{1..m}$. The processing elements are sorted by increasing order of their frequencies ($f_1 \leqslant f_2 \leqslant \ldots \leqslant f_m$). An application A of n tasks is modeled using a DAG graph $G(V, E, w)$. V represents set of nodes in G, and each node $v_i \in V$ represents a task t_i which is characterized by its weight w_i, $i = \overline{1..n}$. We note by W the total sum of the weights $W = \sum_{i=1}^{n} w_i$. E is set of communication edges. Each edge $e_{i,j} \in E$ represents a precedence constraint between two tasks t_i and t_j and refers to the volume of communication from t_i to t_j denoted by $Ct_{i,j}$ if they are not assigned to the same processing element. Communication cost between each pair of processing elements (PE_k, PE_l) is denoted by $Cm_{k,l}$ with $Cm_{k,l} \geqslant Max_i\ execut_{i,k}, \forall i \in \{1, 2, \ldots, n\}$ and $\forall k, l \in \{1, 2, \ldots, m\}$ as in [6].

A task t_i can be executed only after the execution of all its predecessors. We do not allow duplication of tasks or preemption. A task can be executed by all processing units. Execution of task t_i on PE_k generates execution time equal to $execut_{i,k} = \frac{w_i}{f_k}$ and power $p_{i,k} = w_i * f_k^2$. We denote by E the allowed quantity of energy consumed during the execution. E represents in our case an energy bound that should not be exceeded during the execution.

We focus this work to a chain of tasks. Our problem can be modeled by mixed integer quadratic constrained program (P). The first constraint simply expresses that each task must be executed only once and on a single processing element. Constraint (2) keeps energy consumption during execution less than

E. The third constraint describes that the task t_{i+1} must be carried out after the starting time of the task t_i ($i = \overline{1..n-1}$) plus the execution time of t_i. The communication cost ($Ct_{i,i+1} + Cm_{j1,j2}$) is added if both tasks are executed on two different processing elements (PE_{j1} and PE_{j2}) s.t $x_{i,j1} = 1$ and $x_{i+1,j2} = 1$.

$$x_{i,j} = \begin{cases} 1 \text{ if task } t_i \text{ is placed on the processing element} PE_j, i = \overline{1..n},\ j = \overline{1..m} \\ 0 \text{ otherwise} \end{cases}$$

$start_i$ = the starting time of the task $t_i, i = \overline{1..n}$.

$$(P) \begin{cases} \sum_{j=1}^{m} x_{i,j} = 1, \forall i = \overline{1..n} & (1) \\ \sum_{i=1}^{n} \sum_{j=1}^{m} x_{i,j} * p_{i,j} \leq E & (2) \\ start_i + x_{i,j_1} * execut_{i,j_1} + x_{i,j_1} * x_{i+1,j_2}(Ct_{i,i+1} + Cm_{j_1,j_2}) \leqslant start_{i+1} & (3) \\ \qquad \forall j_1 = \overline{1..m}, \quad \forall j_2 = \overline{1..m} \qquad \forall i = \overline{1..n-1} \quad j_1 \neq j_2 & \\ Z(min) = start_n + \sum_{j=1}^{m} x_{n,j} * execut_{n,j} & \end{cases}$$

4 Optimal Scheduling Algorithm for a Chain of Preemptive Tasks

In this section we propose an algorithm to find the optimal solution of the preemptive scheduling without communication cost for a chain of n tasks on a set of m processing elements.

Lemma 1. *The set of schedules that saturate energy constraint is dominant.*

Proof. Let $\widehat{C}_{max}$ be the makespan of a solution such that $\widehat{C}_{max} = \frac{P_1}{f_1} + \frac{P_2}{f_2} + \ldots + \frac{P_m}{f_m}$, $P_i \geqslant 0$ is the quantity of work put on the processing element PE_i, $i = \overline{1..m}$. $\sum_{i=1}^{m} P_i = W$. We assume that $\sum_{j=1}^{m} P_j * f_j^2 < E$. We construct another solution such that: $l = max\{j \in \{1..m\},\ \sum_{i=1}^{j} P_i f_m^2 + \sum_{i=j+1}^{m} P_i f_i^2 < E\}$ and $P_1^{'} = 0$, $P_2^{'} = 0, \ldots, P_l^{'} = 0$, $P_{l+1}^{'} = \frac{E - \sum_{j=1}^{l+1} P_j f_m^2 - \sum_{j=l+2}^{m} P_j f_j^2}{f_{l+1}^2 - f_m^2}$, $P_{l+2}^{'} = P_{l+2}, \ldots, P_m^{'} = P_m + \sum_{j=1}^{l} P_j + (P_{l+1} - P_{l+1}^{'})$. We obtain a new solution $\widehat{C}_{max}^{'} = \sum_{j=1}^{m} \frac{P_j^{'}}{f_j}$ with $\sum_{j=1}^{m} P_j^{'} f_j^2 = E$. $\widehat{C}_{max}^{'} = \sum_{j=1}^{m} \frac{P_j^{'}}{f_j} = \frac{P_{l+1}^{'}}{f_{l+1}} + \sum_{j=l+2}^{m} \frac{P_j}{f_j} + \frac{\sum_{j=1}^{l} P_j + (P_{l+1} - P_{l+1}^{'})}{f_m}$. Since $f_m > f_j$, $j = \overline{1..l+1}$, induces $\frac{P_{l+1}^{'}}{f_{l+1}} + \frac{(P_{l+1} - P_{l+1}^{'})}{f_m} \leqslant \frac{P_{l+1}}{f_{l+1}}$ and $\frac{\sum_{j=1}^{l} P_j}{f_m} \leqslant \sum_{j=1}^{l} \frac{P_j}{f_j}$. Then, we obtain $\sum_{j=1}^{m} \frac{P_j^{'}}{f_j} \leqslant \sum_{j=1}^{m} \frac{P_j}{f_j}$. Finally, $\widehat{C}_{max}^{'} \leqslant \widehat{C}_{max}$. □

Theorem 1. *The following Algorithm 1 gives the optimal solution for preemptive scheduling without communication cost with a complexity of $\theta(m)$.*

We start by finding the fastest processing element PE_j, on which we can perform all the tasks. Then we look for the weight of tasks that can be put on the next processing element (PE_{j+1}) in order to saturate the energy constraint. We denote by W_j the quantity of work put on the processing element PE_j, W_{j+1}

on PE_{j+1}. The best solution is obtained when the energy constraint is saturated s.t $W_j f_j^2 + W_{j+1} f_{j+1}^2 = E$ with $W_j + W_{j+1} = W$. The solution of the system of two equations with two unknowns is $W_j = \frac{E - W * f_{j+1}^2}{f_j^2 - f_{j+1}^2}$ and $W_{j+1} = W - W_j$. This keeps the realizability of the solution: $E - W * f_{j+1}^2 \leqslant 0$ because $W * f_{j+1}^2 \geqslant E$ and $f_j^2 - f_{j+1}^2 < 0$ because $f_j < f_{j+1}$. Then $W \geqslant W_j > 0$ induces $W_{j+1} \geqslant 0$.

Algorithm 1. Preemptive scheduling (PS).

Data: Set of processing elements $M = \{PE_j, j = 1..m\}$ with $f_1 \leqslant f_2 \leqslant \ldots \leqslant f_m$, weights of the tasks $w_1, w_2, \ldots, w_n$, E.
Result: Optimal preemptive scheduling.
begin
  $W = \sum_{i=1}^{n} w_i$; $j = max\{l \in \{1..m\}, W * f_l^2 \leqslant E\}$
  if $W * f_j^2 < E$ **then**
    $W_j = \frac{E - W * f_{j+1}^2}{f_j^2 - f_{j+1}^2}$
    $W_{j+1} = W - W_j$
  else
    $W_j = W$, $W_{j+1} = 0$
  $k = max\{p \in \{1..n\}, \sum_{i=1}^{p} w_i < W_j\}$; $w'_{k+1} = W_j - \sum_{i=1}^{k} w_i$
  Put $t_1 ... t_k$ and a part w'_{k+1} of t_{k+1} on PE_j
  Put $t_{k+2} ... t_n$ and the rest $(w_{k+1} - w'_{k+1})$ of t_{k+1} on PE_{j+1}

We show in the following that Algorithm 1 gives an optimal solution. Let $\widehat{C}_{max}$ be the makespan of the solution obtained by the Algorithm 1: $\widehat{C}_{max} = \frac{W_j}{f_j} + \frac{W_{j+1}}{f_{j+1}}$ due to the precedence constraint. Let $\widehat{C}'_{max} = \frac{P_1}{f_1} + \frac{P_2}{f_2} + \ldots + \frac{P_k}{f_k}$ be another solution on a set of $k > 2$ processing elements, $\sum_{i=1}^{k} P_i = W$. We distinguish three possible cases. The first case corresponds to all frequencies are lower than f_j s.t $f_1 \leqslant f_2 \leqslant \ldots \leqslant f_k \leqslant f_j$. Hence, $\frac{1}{f_i} \geqslant \frac{1}{f_j}$ induces $\frac{P_i}{f_i} \geqslant \frac{P_i}{f_j}$, $\forall\ i = \overline{1..k}$. Follows $\sum_{i=1}^{k} \frac{P_i}{f_i} \geqslant \frac{\sum_{i=1}^{k} P_i}{f_j} = \frac{W}{f_j}$. Finally, since $f_j < f_{j+1}$ induces $\sum_{i=1}^{k} \frac{P_i}{f_i} \geqslant \frac{W}{f_j} \geqslant \frac{W_j}{f_j} + \frac{W_{j+1}}{f_{j+1}}$. Then, $\widehat{C}'_{max} \geqslant \widehat{C}_{max}$.
The second case corresponds to all frequencies are greater than f_{j+1} such that $f_{j+1} \leqslant f_1 \leqslant f_2 \leqslant \ldots \leqslant f_k$. Hence, $\sum_{i=1}^{k} P_i * f_i^2 \geqslant \sum_{i=1}^{k} P_i * f_{j+1}^2 = W * f_{j+1}^2 > E$. The last case corresponds to $f_1 \leqslant \ldots \leqslant f_j < f_{j+1} \leqslant \ldots \leqslant f_k$. To study this case, we start with the following Lemma 2.

Lemma 2. *Let A, B, C be three positive integers such as $1 \leq A < B < C$ and W_1, W_2 be two non negative integers such as $W_1 + W_2 = W$. If $W_1 * A^2 + W_2 * C^2 = W * B^2$ then $\frac{W_1}{A} + \frac{W_2}{C} > \frac{W}{B}$.*

Proof. By replacing W_2 by $(W - W_1)$ in $W_1 * A^2 + W_2 * C^2 = W * B^2$, we obtain $W_1 = W(\frac{C^2 - B^2}{C^2 - A^2})$. Then, by replacing W_1 by $(W - W_2)$ we obtain $W_2 = W(\frac{B^2 - A^2}{C^2 - A^2})$. Follows, $\frac{W_1}{A} + \frac{W_2}{C} = \frac{W(C^2 - B^2)}{A(C^2 - A^2)} + \frac{W(B^2 - A^2)}{C(C^2 - A^2)}$.

Let $\Delta = \frac{W_1}{A} + \frac{W_2}{C} - \frac{W}{B}$, we prove in the following that $\Delta > 0$.
$\Delta = \frac{W(C^2-B^2)}{A(C^2-A^2)} + \frac{W(B^2-A^2)}{C(C^2-A^2)} - \frac{W}{B} = \frac{W}{C^2-A^2}(\frac{C^2-B^2}{A} + \frac{B^2-A^2}{C} - \frac{(C^2-A^2)}{B})$.
We set $X = \frac{B}{A}$ and $Y = \frac{C}{A}$. Observe that $X > 1$ and $Y > X$. Follows:
$\Delta = \frac{W}{Y^2A^2-A^2}(\frac{Y^2A^2-X^2A^2}{A} + \frac{X^2A^2-A^2}{YA} - \frac{(Y^2A^2-A^2)}{XA})$.
$\Delta = \frac{W}{Y^2A-A}(\frac{XY^3-X^3Y+X^3-X-Y^3+Y}{XY}) = \frac{W}{Y^2A-A}(\frac{-(X-1)(Y-1)(X-Y)(X+Y+1)}{XY})$.
Since $X > Y > 1$ we have $(Y-1) > 0$, $(X-1) > 0$ and $(X-Y) < 0$.
Therefore $\frac{-(X-1)(Y-1)(X-Y)(X+Y+1)}{XY} > 0$. Furthermore, $\frac{W}{Y^2A-A} > 0$ because $Y > 1$. Finally, $\Delta > 0$ induces $\frac{W_1}{A} + \frac{W_2}{C} > \frac{W}{B}$. □

Proposition 1. *If* $\sum_{i=1}^{k} P_i * f_i^2 = W_j * f_j^2 + W_{j+1} * f_{j+1}^2$ *and* $\sum_{i=1}^{k} P_i = W_j + W_{j+1}$, *then* $\sum_{i=1}^{k} \frac{P_i}{f_i} > \frac{W_j}{f_j} + \frac{W_{j+1}}{f_{j+1}}$.

Proof. Let φ be a sequence of real such as $\varphi_1 = f_1$ and $\varphi_i = \sqrt{\frac{\sum_{\alpha=1}^{i-1} P_\alpha * \varphi_{i-1}^2 + P_i * f_i^2}{\sum_{\alpha=1}^{i} P_\alpha}}$,
for $i = \overline{2..j}$. This sequence guarantees that $\varphi_{i-1} < \varphi_i < f_i$, $\forall\ i = \overline{2..j}$. Indeed, since $\varphi_1 = f_1$, $\varphi_2^2 = \frac{P_1 * \varphi_1^2 + P_2 * f_2^2}{P_1+P_2} > \frac{P_1 * f_1^2 + P_2 * f_1^2}{P_1+P_2} = f_1^2$.
Furthermore, $\frac{P_1 * f_1^2 + P_2 * f_2^2}{P_1+P_2} < \frac{P_1 * f_2^2 + P_2 * f_2^2}{P_1+P_2} < f_2^2$ induces $\varphi_1 < \varphi_2 < f_2$.
We assume that this is true for $i = j-1$ i.e. $\varphi_{j-2} < \varphi_{j-1} < f_{j-1}$.
$\varphi_j^2 = \frac{\sum_{\alpha=1}^{j-1} P_\alpha * \varphi_{j-1}^2 + P_j * f_j^2}{\sum_{\alpha=1}^{j} P_\alpha} < \frac{\sum_{\alpha=1}^{j-1} P_\alpha * f_{j-1}^2 + P_j * f_j^2}{\sum_{\alpha=1}^{j} P_\alpha} < \frac{\sum_{\alpha=1}^{j-1} P_\alpha * f_j^2 + P_j * f_j^2}{\sum_{\alpha=1}^{j} P_\alpha} = f_j^2$.
$\varphi_j^2 = \frac{\sum_{\alpha=1}^{j-1} P_\alpha * \varphi_{j-1}^2 + P_j * f_j^2}{\sum_{\alpha=1}^{j} P_\alpha} > \frac{\sum_{\alpha=1}^{j-1} P_\alpha * \varphi_{j-1}^2 + P_j * \varphi_j^2}{\sum_{\alpha=1}^{j} P_\alpha}$ induces $\varphi_j^2 > \varphi_{j-1}^2$.
Finally, $\varphi_{j-1} < \varphi_j < f_j$. By recurrence, we deduce that $\varphi_{i-1} < \varphi_i < f_i$.
From Lemma 2 we have:
$\frac{P_1}{f_1} + \frac{P_2}{f_2} > \frac{P_1+P_2}{\varphi_2}, \frac{P_1+P_2}{\varphi_2} + \frac{P_3}{f_3} > \frac{\sum_{i=1}^{3} P_i}{\varphi_3}$ and then, $\forall\ l \in \{1..j\}$ $\frac{\sum_{i=1}^{l-1} P_i}{\varphi_{l-1}} + \frac{P_l}{f_l} > \frac{\sum_{i=1}^{l} P_i}{\varphi_l}$. Follows, $\sum_{i=1}^{j} \frac{P_i}{f_i} > \frac{\sum_{i=1}^{j} P_i}{\varphi_j}$.

Let another sequence of real ϕ such as $\phi_k = f_k$ and $\phi_i = \sqrt{\frac{\sum_{\alpha=i+1}^{k} P_\alpha * \phi_{i+1}^2 + P_i * f_i^2}{\sum_{\alpha=i}^{k} P_\alpha}}$
for $i \in \{j+1..k-1\}$. In the same way, we get $f_i < \phi_i < \phi_{i+1}$, $\forall\ i \in \{j+1..k-1\}$.
And from Lemma 2, we obtain $\sum_{i=j+1}^{k} \frac{P_i}{f_i} > + \frac{\sum_{i=j+1}^{k} P_i}{\phi_{j+1}}$.
It result that $\sum_{i=1}^{k} \frac{P_i}{f_i} > \frac{\sum_{i=1}^{j} P_i}{\varphi_j} + \frac{\sum_{i=j+1}^{k} P_i}{\phi_{j+1}}$.
In order to apply once again Lemma 2, we have to decompose $\sum_{i=1}^{j} P_i$ and $\sum_{i=j+1}^{k} P_i$ into 4 values $W_{L1}, W_{L2}, W_{R1}, W_{R2}$ such that:

$$\begin{cases} W_{L1} + W_{L2} = \sum_{i=1}^{j} P_i \\ W_{R1} + W_{R2} = \sum_{i=j+1}^{k} P_i \\ W_{L1} + W_{R1} = W_j \\ W_{L2} + W_{R2} = W_{j+1} \\ W_{L1} * \varphi_j^2 + W_{R1} * \phi_{j+1}^2 = W_j * f_j^2 \\ W_{L2} * \varphi_j^2 + W_{R2} * \phi_{j+1}^2 = W_{j+1} * f_{j+1}^2 \end{cases} \Longrightarrow \begin{cases} W_{L1} = W_j * \frac{(\phi_{j+1}^2 - f_j^2)}{(\phi_{j+1}^2 - \varphi_j^2)} \\ W_{L2} = W_{j+1} * \frac{(\phi_{j+1}^2 - f_{j+1}^2)}{(\phi_{j+1}^2 - \varphi_j^2)} \\ W_{R1} = W_j * \frac{(f_j^2 - \varphi_j^2)}{(\phi_{j+1}^2 - \varphi_j^2)} \\ W_{R2} = W_{j+1} * \frac{(f_{j+1}^2 - \varphi_j^2)}{(\phi_{j+1}^2 - \varphi_j^2)} \end{cases}$$

This part of proof is illustrated by Fig. 1. Observe that the result values are all

positive. From Lemma 2, we obtain $\frac{W_{L1}}{\varphi_j} + \frac{W_{R1}}{\phi_{j+1}} > \frac{W_j}{f_j}$ and $\frac{W_{L2}}{\varphi_j} + \frac{W_{R2}}{\phi_{j+1}} > \frac{W_{j+1}}{f_{j+1}}$.
Hence $\frac{W_L}{\varphi_j} + \frac{W_R}{\phi_{j+1}} = \frac{W_{L1}}{\varphi_j} + \frac{W_{R1}}{\phi_{j+1}} + \frac{W_{L2}}{\varphi_j} + \frac{W_{R2}}{\phi_{j+1}} > \frac{W_1}{f_j} + \frac{W_2}{f_{j+1}}$.
Follows, $\sum_{i=1}^{k} \frac{P_i}{f_i} > \frac{W_j}{f_j} + \frac{W_{j+1}}{f_{j+1}}$. □

Now, from Proposition 1, $\widehat{C}'_{max} = \sum_{i=1}^{k} \frac{P_i}{f_i} > \widehat{C}_{max} = \frac{W_j}{f_j} + \frac{W_{j+1}}{f_{j+1}}$.

Remark 1. The proof remains valid if $\sum_{i=1}^{k} P_i * f_i^2 \leqslant W_j * f_j^2 + W_{j+1} * f_{j+1}^2$. Indeed, from Lemma 1, we can construct another solution with $P_1^{'}, P_2^{'}, \ldots, P_k^{'}$ such as $\sum_{i=1}^{k} P_i^{'} = \sum_{i=1}^{k} P_i$ and $\sum_{i=1}^{k} P_i^{'} * f_i^2 = W_j * f_j^2 + W_{j+1} * f_{j+1}^2$. Hence, we obtain $\frac{W_j}{f_j} + \frac{W_{j+1}}{f_{j+1}} < \sum_{i=1}^{k} \frac{P_i^{'}}{f_i} < \sum_{i=1}^{k} \frac{P_i}{f_i}$.

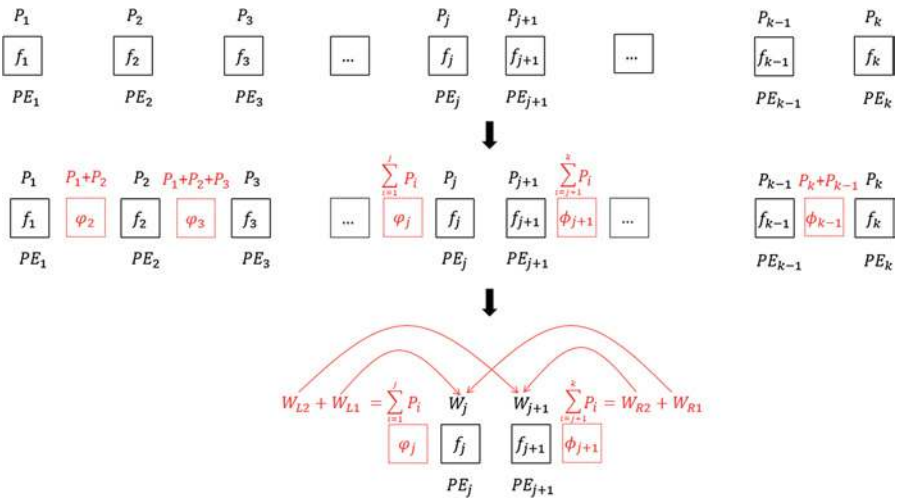

Fig. 1. Resume of the first part of the proof.

5 An Approximation Scheduling Algorithm for Chain of Non-preemptive Tasks with Communication Costs

We assume here a communication cost $Cm_{j,j+1}$ between PE_j and PE_{j+1} and communication cost $Ct_{i,i+1}$ between each pair of tasks t_i and t_{i+1} with $2 * min_i\ Ct_{i,i+1} \geqslant max_j\ Ct_{j,j+1}$, $\forall\ i, j \in \{1..n-1\}$. We do not allow preemption of tasks and we transform the previous solution of preemptive scheduling, using the processing elements PE_j and PE_{j+1} only.

Proposition 2. *If only two processing elements PE_j and PE_{j+1} are available, the schedules with only one communication between them are dominant.*

Proof. Let $\{t_{k+2} \dots t_n\}$ be the set of uncut tasks of the preemptive solution on PE_{j+1} and S_1 the sum of their weights. Let C_{max1} the makespan of a feasible solution obtained by processing tasks $\{t_1 \dots t_{k+1}\}$ on PE_j and $\{t_{k+2} \dots t_n\}$ on PE_{j+1}. By contradiction, let suppose that there exists a feasible solution with at least two displacements such as $S_1 \leqslant S_2$, where S_2 is the sum of the tasks weights on PE_{j+1} with this solution, let C_{max2} be its makespan. We prove that $C_{max2} \geqslant C_{max1}$. Since the second solution is feasible, $S2 \leqslant W_{j+1}$. By the previous algorithm, $W_{j+1} \leqslant S_1 + w_{k+1} \leqslant S_1 + max\ w_i$ with $i \in \{1 \dots n\}$, and thus $S_2 \leqslant S_1 + \text{Max } w_i, i = \overline{1..n}$. $C_{max1} = \frac{W - S_1}{f_j} + \frac{S_1}{f_{j+1}} + Cm_{j,j+1} + Ct_{k+1,k+2}$ and $C_{max2} \geqslant \frac{W - S_2}{f_j} + \frac{S_2}{f_{j+1}} + 2 * Cm_{j,j+1} + 2 * min\ Ct_{i,i+1}$, $i \in \{1 \dots n-1\}$. Follows, $C_{max2} - C_{max1} = \frac{S_2 - S_1}{f_{j+1}} - \frac{S_2 - S_1}{f_j} + Cm_{j,j+1} + 2 * min\ Ct_{i,i+1} - Ct_{k+1,k+2}$. Since $\frac{S_2 - S_1}{f_{j+1}} \geqslant 0$ and $2 * min\ Ct_{i,i+1} - Ct_{k+1,k+2} \geqslant 0, \forall\ i = \overline{1..n-1}$, induce $C_{max2} - C_{max1} \geqslant Cm_{j,j+1} - \frac{S_2 - S_1}{f_j}$. Finally, $Cm_{j,j+1} - \frac{S_2 - S_1}{f_j} \geqslant Cm_{j,j+1} - \frac{Max\ w_i}{f_j}$, $i = \overline{1..n}$. According to the hypothesis, $Cm_{j,j+1} - \text{Max } \frac{w_i}{f_j} \geqslant 0, \forall\ i = \overline{1..n}$. Therefore $C_{max2} - C_{max1} \geqslant 0 \Longrightarrow C_{max2} \geqslant C_{max1}$. □

Theorem 2. *The following Algorithm 2 provides a solution for non-preemptive scheduling starting from the preemptive scheduling solution obtained by Algorithm 1 with a complexity of* $\theta(n + m)$.

The two variables α and β are used to determine the assignment of tasks. In the case $W_{j+1} = 0$, we put all the tasks on PE_j. Otherwise, let $Cost_1(v)$ be the cost of executing the first tasks (t_1 to t_v) on PE_j with $\sum_{i=1}^{v} w_i \geqslant W_j$, then the rest on PE_{j+1}. $Cost_1(v) = \{Ct_{v,v+1} + \frac{\sum_{i=1}^{v} w_i}{f_j} + \frac{\sum_{i=v+1}^{n} w_i}{f_{j+1}} + Cm_{j,j+1}\}$. Let $Cost_2(v)$ be the cost of executing the first tasks (t_1 to t_v) on PE_{j+1}, then the rest on PE_j with $\sum_{i=v+1}^{n} w_i \geqslant W_j$. $Cost_2(v) = \{Ct_{v,v+1} + \frac{\sum_{i=1}^{v} w_i}{f_{j+1}} + \frac{\sum_{i=v+1}^{n} w_i}{f_j} + Cm_{j,j+1}\}$.

We start by finding the tasks v_1 and v_2 that give the best respective scheduling makespan ($Cost_1$) and ($Cost_2$), and keeping the best one. Finally, we check if the cost generated by using both processing elements PE_j and PE_{j+1} is less than the scheduling makespan obtained by performing all tasks on PE_j.

Example 1. Consider the task graph given by Figure 2. It contains ten task nodes ($n = 10$) labeled from t_1 to t_{10} with two additional nodes S and E (beginning and end of the application). The edges are labeled with the communication cost between tasks. The nodes are labeled with the weight of each task.
Consider a heterogeneous platform with 3 processing elements, their frequencies are given in Table 1. The communication cost between processing elements are given in Table 2. The maximum energy consumption is $E = 1350$.

The application of preemptive scheduling Algorithm 2 gives $PE_j = PE_2$ and $PE_{j+1} = PE_3$ with $W_2 = 0,5625$ and $W_3 = 37,4375$. Since $W_3 > 0$, we obtain $Cost_1 = Ct_{1,2} + \frac{w_1}{f_2} + \frac{\sum_{i=2}^{10} w_i}{f_3} + Cm_{2,3} = 17$, $v_1 = 1$. $Cost_2 = Ct_{7,8} + \frac{\sum_{i=1}^{7} w_i}{f_2} + \frac{\sum_{i=8}^{10} w_i}{f_3} + Cm_{2,3} = 19$, $v_2 = 7$. $Cost_1 < Cost_2$, induces $Cost = Cost_1 = 17$, $\beta = 1$ and $\alpha = 1$. Finally, $\frac{W}{f_2} = \frac{38}{2} = 19 > Cost$. We put the task t_1 on the

Algorithm 2. Non-Preemptive Scheduling (NPS).

Data: Weights of the tasks $w_1, w_2, \ldots, w_n$. Communication costs between tasks $Ct_{i,i+1}$, $i \in \{1..n-1\}$.

Result: Approximate solution $\widehat{C}_{max}$ for non preemptive scheduling.

begin

Find PE_j, PE_{j+1} and W_j, W_{j+1} with preemptive scheduling algorithm

if $W_j = W$ **then**

$\beta = n$, $\alpha = 1$

else

$Cost_1 = min\{Cost_1(v), v \in \{1..n-1\}, \sum_{i=1}^{v} w_i \geqslant W_j\}$; let $v_1 \in \{1..n-1\}$ such that $Cost_{v_1} = Cost_1$

$Cost_2 = min\{Cost_2(v), v \in \{1..n-1\}, \sum_{i=v_2+1}^{n} w_i \geqslant W_j\}$; let $v_2 \in \{1..n-1\}$ such that $Cost_{v_2} = Cost_2$

if $Cost_1 < Cost_2$ **then**

$Cost = Cost_1$, $\beta = v_1$, $\alpha = 1$

else

$Cost = Cost_2$, $\beta = n$, $\alpha = v_2 + 1$

if $Cost > \frac{W}{f_j}$ **then**

$Cost = \frac{W}{f_j}$, $\beta = n$, $\alpha = 1$

Put tasks between t_α and t_β on the processing element PE_j

Order the rest on the processing element PE_{j+1}, $\widehat{C}_{max}$= $Cost$

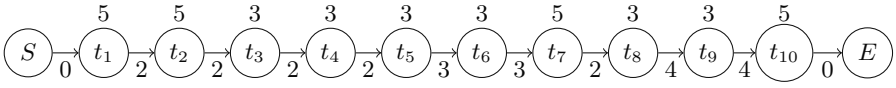

Fig. 2. Task chain graph.

Table 1. Frequencies of processing elements.

PE_j	PE_1	PE_2	PE_3
f_j	1	2	6

Table 2. Communication cost between processing elements.

PE_j	PE_1	PE_2	PE_3
PE_1	0	7	6
PE_2	7	0	7
PE_3	6	7	0

processing element PE_2 and tasks t_2 to t_{10} on PE_3. $\widehat{C}_{max} = Cost = 17$. For this instance, our approach gives the optimal solution.

Proposition 3. *Let $C^{\star}_{max}$ be the optimal solution for non-preemptive scheduling and $\widehat{C}_{max}$ the solution obtained by Algorithm* 2, then $\frac{\widehat{C}_{max}}{C^{\star}_{max}} \leqslant \frac{W}{W_j + \frac{f_j W_{j+1}}{f_{j+1}}}$.

Proof. The optimal solution C'_{max} of the preemptive scheduling is given by $C'_{max} = \frac{W_j}{f_j} + \frac{W_{j+1}}{f_{j+1}}$. In the worst case for our algorithm, all tasks are

executed on the processing element f_j, thus we get $\widehat{C}_{max} \leqslant \frac{W}{f_j}$. Follows $\frac{\widehat{C}_{max}}{C'_{max}} \leqslant \frac{\frac{W}{f_j}}{\frac{W_j}{f_j}+\frac{W_{j+1}}{f_{j+1}}} \leqslant \frac{W}{W_j+\frac{f_j W_{j+1}}{f_{j+1}}}$. By optimality of Algorithm 1, $C'_{max} \leqslant C^{\star}_{max}$ induces $\frac{\widehat{C}_{max}}{C^{\star}_{max}} \leqslant \frac{\widehat{C}_{max}}{C'_{max}}$, then $\frac{\widehat{C}_{max}}{C^{\star}_{max}} \leqslant \frac{W}{W_j+\frac{f_j W_{j+1}}{f_{j+1}}}$. □

Remark 2. This ratio is reached, let consider an instance which generates $W_{j+1} = 0$ and $W_j = W$ for the preemptive solution, then, $1 \leqslant \frac{\widehat{C}_{max}}{C^{\star}_{max}} \leqslant \frac{W}{W_j+\frac{f_j W_{j+1}}{f_{j+1}}} = \frac{W}{W} = 1$. So, we obtain the optimal solution, $\widehat{C}_{max} = C^{\star}_{max}$.

Remark 3. Since $\frac{f_j}{f_{j+1}} < 1$, $\frac{W}{W_j+\frac{f_j W_{j+1}}{f_{j+1}}} < \frac{W}{\frac{f_j}{f_{j+1}}(W_j+W_{j+1})} = \frac{f_{j+1}}{f_j}$, and finally, $\frac{\widehat{C}_{max}}{C^{\star}_{max}} < \frac{f_{j+1}}{f_j}$.

6 Experimental Results

In order to measure the efficiency of our algorithm, we performed several tests on randomly generated instances with different dimensions. For this purpose, we developed a random instances generator in *C++* adjustable with several parameters.

General settings are number of tasks n and processing elements m. We denote by *test_n_m* instance defined by these two parameters. The weights of the tasks are generated randomly over an interval $[w_{min}, w_{max}]$. The frequencies of the processing elements are randomly generated over an interval $[f_{min}, f_{max}]$ while ensuring the heterogeneity of the system by generating different values. The communication costs between tasks are generated randomly over an interval $[Ct_{min}, Ct_{max}]$ and between processing elements over an interval $[Cm_{min}, Cm_{max}]$ in accordance with the hypothesis described in Sect. 3. The bound E is randomly generated with respect to $W * f_1^2 < E < W * f_m^2$.

Our proposed Algorithm 2 were implemented in *C++*. The exact solution is obtained by solving the model (P) with *CPLEX* 12.5.0 [4] and the OPL script language. The following Table 3 shows the results of tests on different instance sizes. We have generated 30 instances for the first four rows (from instance *test_8_3* to *test_20_4*) and then one instance for the others due to the large running time on *CPLEX*.

The PS (Preemptive Scheduling) columns present the makespan average solution obtained by the Algorithm 1. The NPS (Non-Preemptive Scheduling) columns present the makespan average solution obtained by the Algorithm 2 and its average execution time. The *CPLEX* columns present the average makespan solution of the resolution of the model (P) with *CPLEX* and the average computation cost required as well as the optimality of the solutions. Finally, the columns GAP_1 and GAP_2 present the average ratio between the solution obtained by $Bound_1 = CPLEX$ solution and $Bound_2 =$ Preemtive solution with NPS solutions which is calculated as follow:

$$GAP_i = \frac{\text{Heuristic Solution} - Bound_i}{Bound_i} * 100, i \in \{1, 2\}.$$

Since the execution time of a quadratic model is generally too large, we have therefore limited the running time for *CPLEX* to 60 min. In Table 3, we can notice that for most of the instances with less than 30 tasks, our algorithm gives an optimal solution with smaller running time than *CPLEX*. Moreover, for larger instances, *CPLEX* takes much longer to find a solution, whereas NPS gives a solution in less than one second for an instance with 10000 tasks.

Table 3. Evaluation of the NPS heuristic compared to *CPLEX*.

Instances	PS	NPS		*CPLEX*			Gap_1	Gap_2
		Sol	Time	Sol	Time	Opt		
test_8_3	15.03	19.89	0.0004 s	19.89	0.76 s	X	0%	31.14%
test_12_3	24.82	32.55	0.0005 s	32.55	2.17 s	X	0%	15.80%
test_15_4	29.87	34.59	0.0005 s	34.58	6.46 s	X	0.02%	21.91%
test_20_4	27.51	33.54	0.0007 s	33.54	8 min 3 s	X	0%	2.25%
test_30_6	24.84	25.40	0.001 s	25.40	35 min	X	0%	7.02%
test_50_6	40.27	43.10	0.01 s	102.24	60 min	/	−57.84%	8.03%
test_100_9	53.78	55.39	0.02 s	843.48	60 min	/	−93.43%	2.99%
test_200_9	123.21	128.92	0.03 s	1929.20	60 min	/	−93.31%	4.63%
test_500_11	207.44	217.52	0.05 s	3587.39	60 min	/	−93.93%	4.85%
test_10000_11	4814.62	4828.06	0.5 s	/	60 min	/	/	0.27%

7 Conclusion and Future Work

This paper presents an efficient approximation algorithm to solve the task scheduling problem on heterogeneous platform for the particular case of linear chain of tasks. Our objective is to minimize both the total execution time (makespan) and the energy consumption by imposing a constraint on the total energy consumed by the system. This work has shown that finding an efficient scheduling is not easy. Tests on large instances close to reality shows the limits of solving the problem with a solver such as *CPLEX*.

The main contribution of this work is to give an algorithm which provides a solution with small running time, and also guarantee the quality of the solution obtained compared to the optimal solution. The ratio obtained depends on the frequencies of two successive processing elements PE_j and PE_{j+1} used in preemptive scheduling. The performance ratio of our algorithm is bounded by $\frac{f_{j+1}}{f_j}$. As part of the future, we will focus on the extension to more general classes of graphs to handle real application.

References

1. Aupy, G., Benoit, A., Dufossé, F., Robert, Y.: Reclaiming the energy of a schedule: models and algorithms. Concur. Comput.: Pract. Exp. **25**(11), 1505–1523 (2013)
2. Fard, H.M., Prodan, R., Barrionuevo, J.J.D., Fahringer, T.: A multi-objective approach for workflow scheduling in heterogeneous environments. In: Proceedings of the 2012 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID 2012), pp. 300–309. IEEE Computer Society (2012)
3. Griessl, R., Peykanu, M., Hagemeyer, J., Porrmann, M., Krupop, S., Kosmann, L., Knocke, P., Kierzynka, M., Oleksiak, A., et al.: FPGA-accelerated heterogeneous hyperscale server architecture for next-generation compute clusters (2015)
4. IBM: IBM ILOG CPLEX V12.5 user's manual for CPLEX (2013). http://www.ibm.com
5. Lee, Y.C., Zomaya, A.Y.: Minimizing energy consumption for precedence-constrained applications using dynamic voltage scaling. In: 9th IEEE/ACM International Symposium on Cluster Computing and the Grid, CCGRID 2009, pp. 92–99. IEEE (2009)
6. Zaourar, L., Ait Aba, M., Briand, D., Philippe, J.M.: Modeling of applications and hardware to explore task mapping and scheduling strategies on a heterogeneous micro-server system (2017, to appear in IPDPSW)
7. Mezmaz, M., Melab, N., Kessaci, Y., Lee, Y.C., Talbi, E.G., Zomaya, A.Y., Tuyttens, D.: A parallel bi-objective hybrid metaheuristic for energy-aware scheduling for cloud computing systems. J. Parallel Distrib. Comput. **71**(11), 1497–1508 (2011)
8. Sheikh, H.F., Ahmad, I.: Efficient heuristics for joint optimization of performance, energy, and temperature in allocating tasks to multi-core processors. In: 2014 International Green Computing Conference (IGCC), pp. 1–8. IEEE (2014)
9. Tarplee, K.M., Friese, R., Maciejewski, A.A., Siegel, H.J.: Efficient and scalable pareto front generation for energy and makespan in heterogeneous computing systems. In: Fidanova, S. (ed.) Recent Advances in Computational Optimization. SCI, vol. 580, pp. 161–180. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-12631-9_10
10. Tarplee, K.M., Friese, R., Maciejewski, A.A., Siegel, H.J., Chong, E.K.: Energy and makespan tradeoffs in heterogeneous computing systems using efficient linear programming techniques. IEEE Trans. Parallel Distrib. Syst. **27**(6), 1633–1646 (2016)
11. Vasquez Perez, O.C.: Ordonnancement de tâches pour concilier la minimisation de la consommation d'énergie avec la qualité de service: optimisation et théorie des jeux. Ph.D. thesis, Paris 6 (2014)
12. Xie, G., Xiao, X., Li, R., Li, K.: Schedule length minimization of parallel applications with energy consumption constraints using heuristics on heterogeneous distributed systems. Concurr. Comput.: Pract. Exp. (2016)
13. Young, B.D., Pasricha, S., Maciejewski, A.A., Siegel, H.J., Smith, J.T.: Heterogeneous makespan and energy-constrained DAG scheduling. In: Proceedings of the 2013 Workshop on Energy Efficient High Performance Parallel and Distributed Computing, pp. 3–12. ACM (2013)
14. Zhang, L., Li, K., Li, C., Li, K.: Bi-objective workflow scheduling of the energy consumption and reliability in heterogeneous computing systems. Inf. Sci. **379**, 241–256 (2017)

15. Zhang, L., Li, K., Xu, Y., Mei, J., Zhang, F., Li, K.: Maximizing reliability with energy conservation for parallel task scheduling in a heterogeneous cluster. Inf. Sci. **319**, 113–131 (2015)
16. Zhong, X., Xu, C.Z.: Energy-aware modeling and scheduling for dynamic voltage scaling with statistical real-time guarantee. IEEE Trans. Comput. **56**(3), 358–372 (2007)