On the performance of greedy algorithms for energy minimization

Anne Benoit, Paul Renaud-Goud and Yves Robert

LIP, ENS Lyon, 46 Allée d'Italie, 69364 Lyon Cedex 07, France
UMR 5668 - CNRS - ENS Lyon - UCB Lyon - INRIA
{Anne.Benoit|Paul.Renaud-Goud|Yves.Robert}@ens-lyon.fr

September 2010

LIP Research Report RR-2010-27

Abstract

We revisit the well-known greedy algorithm for scheduling independent jobs on parallel processors, with the objective of energy minimization. We assess the performance of the online version, as well as the performance of the offline version, which sorts the jobs by non-increasing size before execution. We derive new approximation factors, as well as examples that show that these factors cannot be improved, thereby completely characterizing the performance of the algorithms.

Contents

1 Introduction **3**

2 Main results **5**

2.1 Proof of the main theorems 5

2.2 The approximation factor as a function of p 10

3 Conclusion **11**

1 Introduction

This short communication aims at characterizing the performance of the well known greedy algorithm for scheduling independent jobs on parallel processors, when the objective is to minimize the energy consumption instead of the execution time, or makespan.

For convenience, here is a quick background on the greedy algorithm for makespan minimization. Consider a set $\mathfrak{J}$ of n independent jobs $J_1, \ldots, J_n$ to be scheduled on a set $\mathfrak{P}$ of p parallel processors $\mathcal{P}_1, \ldots, \mathcal{P}_p$. Let a_i be the size of job J_i, that is the time it requires for execution. The algorithm comes in two versions, online and offline, or without/with sorting jobs. In the online version of the problem, jobs arrive on the fly. The OnLine-Greedy algorithm assigns the last incoming job to the currently least loaded processor. In the offline version of the problem (see [4]), all job sizes are known in advance, and the OffLine-Greedy starts by sorting the jobs (largest sizes first). Then it assigns jobs to processors exactly as in the online version. The performance of both versions is characterized by the following propositions (see Figures 1 and 2 for an illustration of the worst-case scenarios):

Theorem 1 *For makespan minimization,* OnLine-Greedy *is a* $2-\frac{1}{p}$ *approximation, and this approximation factor is met on the following instance:* $n = p(p-1)+1$, $a_i = 1$ *for* $1 \leq i \leq p(p-1)$ *and* $a_n = p$.

Theorem 2 *For makespan minimization,* OffLine-Greedy *is a* $\frac{4}{3} - \frac{1}{3p}$ *approximation, and this approximation factor is met on the following instance:* $n = 2p+1$, $a_{2i-1} = a_{2i} = 2p-i$ *for* $1 \leq i \leq p$ *and* $a_n = p$.

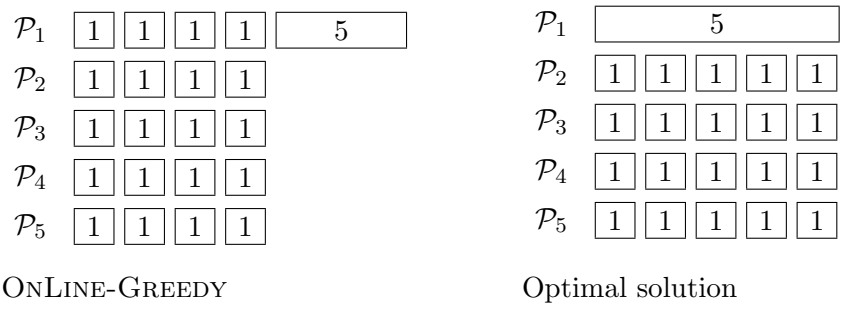


Figure 1: Tight instance for OnLine-Greedy (with $p = 5$).

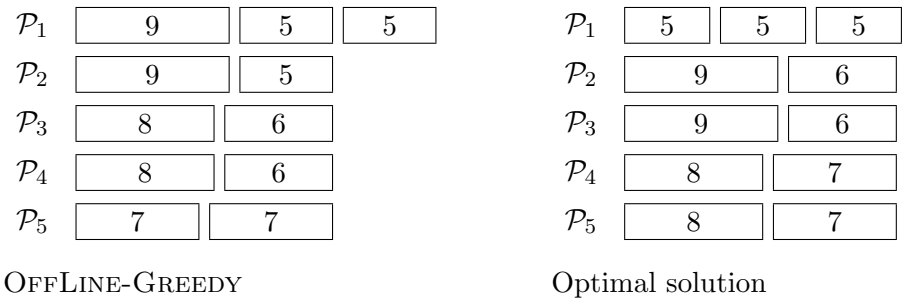


Figure 2: Tight instance for OffLine-Greedy (with $p = 5$).

Minimizing the total energy consumed by the processors to execute the job has recently become a very important objective, both for economic and environmental reasons [7]. Assume

that we can vary processor speeds, for instance through dynamic voltage scaling. In that case we can always use the smallest available speed for each processor, at the price of a dramatic decrease in performance. The problem is in fact a bi-criteria problem: given a bound M on the makespan, what is the schedule that minimizes energy while enforcing the execution time bound? For simplicity, we can assume that processors have continuous speeds [5, 6, 8, 9], and scale the problem instance so that $M = 1$. This amounts to set each processor speed equal to its workload, and to minimize the total energy dissipated during an execution of length one time-unit. In other words, this amounts to minimize the total dissipated power, which is proportional to the sum of the cubes of the processor speeds (a model commonly used, e.g. in [9, 2, 1, 3]).

Formally, let $alloc : \mathfrak{J} \to \mathfrak{P}$ denote the allocation function, and let $load(q) = \{i \mid alloc(J_i) = \mathcal{P}_q\}$ be the index set of jobs assigned to processor $\mathcal{P}_q$ for $1 \leq q \leq p$. The power dissipated by $\mathcal{P}_q$ is $\sum_{i \in load(q)} a_i^3$, hence the objective is to minimize

$$\sum_{q=1}^{p} \sum_{i \in load(q)} a_i^3 \, . \tag{1}$$

This is to be contrasted with the makespan minimization objective, which writes

$$\max_{1 \leq q \leq p} \sum_{i \in load(q)} a_i \, . \tag{2}$$

However, because of the convexity of the cubic power function, the "natural" greedy algorithm is the same for both objectives: assigning the next job to the currently least loaded processor minimizes, among all possible assignments for that job, both the current makespan and dissipated power. We observe that when $p = 2$, the optimal solution is the same for both objectives. However, this is not true for larger values of p. For example, consider the instance with $n = 6$, $p = 3$, $a_1 = 8.1$, $a_2 = a_3 = 5$, $a_4 = a_5 = 4$ and $a_6 = 2$. The optimal solution for the makespan is the partition $\{J_1\}, \{J_2, J_3\}, \{J_4, J_5, J_6\}$, with makespan 10 and power 2531.41. The optimal solution for the power is the partition $\{J_1, J_6\}, \{J_2, J_4\}, \{J_3, J_5\}$, with makespan 10.1 and power 2488.301.

Just as the original makespan minimization problem, the (decision version of the) power minimization problem is NP-complete, and a FPTAS (fully polynomial-time approximation scheme) can be derived. However, the greedy algorithm plays a key role in all situations where jobs arrive on the fly, or when the scheduling cost itself is critical. This was already true for the makespan problem, but may be even more important for the power problem, due to the environmental (or "green") computing perspective that applies to all application fields and computing platforms.

The main results of the paper are summarized in Theorems 3 and 4 below:

Theorem 3 *For power minimization,* OnLine-Greedy *is a $f_p^{(\mathrm{on})}(\beta_p^{(\mathrm{on})})$ approximation, where*

$$f_p^{(\mathrm{on})}(\beta) = \frac{\frac{1}{p^3}\left((1 + (p-1)\beta)^3 + (p-1)\,(1-\beta)^3\right)}{\beta^3 + \frac{((1-\beta)^3}{(p-1)^2}} \, ,$$

and where $\beta_p^{(\mathrm{on})}$ is the unique root in the interval $[\frac{1}{p}, 1]$ of the polynomial

$$g_p^{(\mathrm{on})}(\beta) = \beta^4(-p^3 + 4p^2 - 5p + 2) + \beta^3(-2p^2 + 6p - 4) + \beta^2(-4p + 5) + \beta(2p - 4) + 1.$$

This approximation factor cannot be improved.

Theorem 4 *For power minimization,* OffLine-Greedy *is a* $f_p^{(\mathrm{off})}(\beta_p^{(\mathrm{off})})$ *approximation, where*

$$f_p^{(\mathrm{off})}(\beta) = \frac{\frac{1}{p^3}\left(\left(1+\frac{(p-1)\beta}{3}\right)^3 + (p-1)\left(1-\frac{\beta}{3}\right)^3\right)}{\beta^3 + \frac{(1-\beta)^3}{(p-1)^2}},$$

and where $\beta_p^{(\mathrm{off})}$ *is the unique root in the interval* $[\frac{1}{p}, 1]$ *of the polynomial*

$$g_p^{(\mathrm{off})}(\beta) = \beta^4(-9p^3+30p^2-27p+6)+\beta^3(-6p^2+18p-12)+\beta^2(-78p^2+126p+33)+\beta(18p-180)+81.$$

This approximation factor cannot be improved.

Section 2 is devoted to a detailed proof of both theorems, and also provides numerical values of the approximation factors for small values of p. We give some final remarks in Section 3.

2 Main results

2.1 Proof of the main theorems

The proof of Theorems 3 and 4 is organized as follows:

- Proposition 1 provides a technical bound that is valid for both the online and offline versions;
- This technical is bound is used in Proposition 2 to show that OnLine-Greedy is a $f_p^{(\mathrm{on})}(\beta_p^{(\mathrm{on})})$ approximation, and in Proposition 3 to show that OffLine-Greedy is a $f_p^{(\mathrm{off})}(\beta_p^{(\mathrm{off})})$ approximation;
- Then instances showing that the above factors are tight are given in Proposition 4 for OnLine-Greedy, and in Proposition 5 for OffLine-Greedy.

Proposition 1 *For any given instance, the performance ratio* $\frac{P_{\mathrm{greedy}}}{P_{\mathrm{opt}}}$ *of the greedy algorithm (*OnLine-Greedy *or* OffLine-Greedy*) is such that*

$$\frac{P_{\mathrm{greedy}}}{P_{\mathrm{opt}}} \leq \frac{\left(\frac{S+(p-1)a_j}{p}\right)^3 + (p-1)\left(\frac{S-a_j}{p}\right)^3}{O^3 + (p-1)\left(\frac{S-O}{p-1}\right)^3}, \tag{3}$$

where

- P_{greedy} *is the power dissipated by the greedy algorithm;*
- P_{opt} *is the power dissipated in the optimal solution;*
- $S = \sum_{i=1}^{n} a_i$*;*
- O *is the largest processor load in the optimal solution;*
- j *is the index of the last job assigned to the processor that has the largest load in the greedy algorithm.*

Proof 1 *For the optimal solution, we immediately have*

$$P_{\mathrm{opt}} \geq O^3 + (p-1)\left(\frac{S-O}{p-1}\right)^3.$$

This is because of the definition of O*, and of the convexity of the power function.*

There remains to show that for the greedy algorithm,

$$P_{\text{greedy}} \leq \left(\frac{S+(p-1)a_j}{p}\right)^3 + (p-1)\left(\frac{S-a_j}{p}\right)^3 .$$

Without loss of generality, let $\mathcal{P}_1$ be the maximum loaded processor in the solution returned by the greedy algorithm. For all $q \in \{1,\ldots,p\}$, let M_q be the load of processor $\mathcal{P}_q$ before the assignment of the job J_j, and let $u_q \geq 0$ be the sum of the sizes of all jobs assigned to $\mathcal{P}_q$ after J_{j-1}. By definition of j, we have $u_1 = a_j$. The power returned by the greedy algorithm is thus:

$$P_{\text{greedy}} = \sum_{q=1}^{p}(M_q+u_q)^3 = (M_1+a_j)^3 + \sum_{q=2}^{p}(M_q+u_q)^3 .$$

For $q \in \{2,\ldots,p\}$, let $v_q = M_q - u_q - \frac{S-M_1-a_j}{p-1}$, so that

$$P_{\text{greedy}} = (M_1+a_j)^3 + \sum_{q=2}^{p}(\frac{S-M_1-a_j}{p-1}+v_q)^3 = f(M_1) .$$

Note that the v_q can be either positive or negative. Given the v_q, we have for $p \geq 2$:

$$\begin{aligned} f'(M_1) &\geq \frac{3}{p-1}\sum_{q=2}^{p}\left((M_1+a_j)^2 - \left(\frac{S-M_1-a_j}{p-1}+v_q\right)^2\right) \\ &\geq \frac{3}{p-1}\times\sum_{q=2}^{p}\left(M_1+a_j-\frac{S-M_1-a_j}{p-1}-v_q\right)\times\left(M_1+a_j+\frac{S-M_1-a_j}{p-1}+v_q\right) . \end{aligned}$$

By construction, $M_1+a_j \geq (S-M_1-a_j)/(p-1)+v_q$, therefore f is an increasing function. Moreover, $\mathcal{P}_1$ is the least loaded processor before the assignment of J_j, thus a fortiori, for $q \in \{2,\ldots,p\}$, $M_1 \leq \frac{S-M_1-a_j}{p-1}+v_q$, hence $(p-1)M_1+M_1 \leq (S-M_1-a_j)+\sum_{q=2}^{p}v_q+M_1$. We derive that $M_1 \leq (S-a_j+\sum_{q=2}^{p}v_q)/p = M_1^+$. But since $(M_1+a_j)+\sum_{q=2}^{p}\left(\frac{S-M_1-a_j}{p-1}+v_q\right) = S$, we obtain $\sum_{q=2}^{p}v_q = 0$. Hence M_1^+ does not depend on the v_q, and $M_1^+ = (S-a_j)/p$. Finally $P_{\text{greedy}} = f(M_1) \leq f(M_1^+)$.

We had for $q \in \{2,\ldots,p\}$, $M_1 \leq (S-M_1-a_j)/(p-1)+v_q$, hence if $M_1 = M_1^+$, then $v_q \leq M_1^+ - (S-a_j)/p = 0$. We deduce that for $M_1 = M_1^+$ and $q \in \{2,\ldots,p\}$, $v_q = 0$. We have shown that

$$P_{\text{greedy}} \leq \left(\frac{S+(p-1)a_j}{p}\right)^3 + (p-1)\left(\frac{S-a_j}{p}\right)^3 ,$$

which leads to the desired result.

Proposition 2 *For power minimization,* OnLine-Greedy *is a $f_p^{(\text{on})}(\beta_p^{(\text{on})})$ approximation.*

Proof 2 *We use the notations of Proposition 1. Firstly, we observe that $a_i \leq O$, for each $i \in \{1,\ldots,n\}$, by definition of O. In particular, $a_j \leq O$. We introduce $\beta = \frac{O}{S}$. Clearly, $\beta \in [1/p,1]$, and we can rewrite Equation (3) as:*

$$\frac{P_{\text{online}}}{P_{\text{opt}}} \leq f_p^{(\text{on})}(\beta) = \frac{\frac{1}{p^3}\left((1+(p-1)\beta)^3+(p-1)(1-\beta)^3\right)}{\beta^3+\frac{((1-\beta)^3}{(p-1)^2}} .$$

We now show that, for all p, $f_p^{(\text{on})}$ has a single maximum in $[1/p, 1]$. After differentiating and eliminating some positive multiplicative factor, we obtain that the sign of $\left(f_p^{(\text{on})}\right)'$ is that of $g_p^{(\text{on})}$, where:

$$g_p^{(\text{on})}(\beta) = \beta^4(-9p^3+30p^2-27p+6)+\beta^3(-6p^2+18p-12)+\beta^2(-78p^2+126p+33)+\beta(18p-180)+81.$$

Differentiating again two times, we obtain:

$$\left(g_p^{(\text{on})}\right)'(\beta) = 4\beta^3(-9p^3+30p^2-27p+6)+3\beta^2(-6p^2+18p-12)+2\beta(-78p^2+126p+33)+18p-180\ ;$$

$$\left(g_p^{(\text{on})}\right)''(\beta) = 24\beta^2 - 24\beta + 10 - 8p + p(-12\beta p + 36\beta - 60\beta^2) - p^2(12\beta^3 + 48\beta^2)\ .$$

If $p \geq 5$, $\left(g_p^{(\text{on})}\right)''(\beta) \leq 34 - 40 + p(-60+36) + p^2(-60+48)\beta^2 \leq 0$. We check that $\left(g_2^{(\text{on})}\right)''(\beta) = -6 \leq 0$, $\left(g_3^{(\text{on})}\right)''(\beta) = -24\beta - 14 - 48\beta^2 \leq 0$ and $\left(g_4^{(\text{on})}\right)''(\beta) = -72\beta - 22 - 216\beta^2 \leq 0$, hence $\left(g_p^{(\text{on})}\right)'$ is a decreasing function for all $p \geq 2$ in the interval $[1/p, 1]$.

Next, we show that $\left(g_p^{(\text{on})}\right)'(1) = -4p^3 + 10p^2 - 8p + 2 \leq 0$. Indeed if $p \geq 3$, then $\left(g_p^{(\text{on})}\right)'(1) \leq p^2(-12+10) - 24 + 2 \leq 0$, and $\left(g_2^{(\text{on})}\right)'(2) = -6 \leq 0$. Hence, either $g_p^{(\text{on})}$ is increasing and then decreasing in the interval $[1/p, 1]$, or $g_p^{(\text{on})}$ is decreasing in the whole interval. But we have that $g_p^{(\text{on})}(1) = -p + 2p^2 - p^3 \leq 0$ for $p \geq 2$. Also, $g_p^{(\text{on})}(1/p) = 3 - 11/p + 15/p^2 - 9/p^3 + 2/p^4 \geq 0$, since $g_2^{(\text{on})}(1/2) = 1/4$, $g_3^{(\text{on})}(1/3) = 56/81$ and for $p \geq 4$, $g_p^{(\text{on})}(1/p) \geq 1/p(12-11)$. In both cases ($g_p^{(\text{on})}$ increasing then decreasing, or $g_p^{(\text{on})}$ only decreasing), we conclude that $g_p^{(\text{on})}$ has a single zero $\beta_p^{(\text{on})}$ in $[1/p, 1]$, for which $f_p^{(\text{on})}$ attains its maximum. Finally OnLine-Greedy *is a $f_p^{(\text{on})}(\beta_p^{(\text{on})})$ approximation.*

Proposition 3 *For power minimization,* OffLine-Greedy *is a $f_p^{(\text{off})}(\beta_p^{(\text{off})})$ approximation.*

Proof 3 *We follow the same line of reasoning as in Proposition 2, with $O = \beta S$, but we now further assume that $a_j \leq O/3$. Indeed, if $a_j > O/3$, there are at most two jobs assigned to each processor in the optimal solution. But then $n \leq 2p$, and for all such instances* OffLine-Greedy *is optimal (this is the same argument as for the makespan minimization problem, due to the convexity of the power function). With $a_j \leq O/3 = \beta S/3$, we rewrite Equation (3) as:*

$$\frac{P_{\text{online}}}{P_{\text{opt}}} \leq f_p^{(\text{off})}(\beta) = \frac{\frac{1}{p^3}\left(\left(1 + \frac{(p-1)\beta}{3}\right)^3 + (p-1)\left(1 - \frac{\beta}{3}\right)^3\right)}{\beta^3 + \frac{(1-\beta)^3}{(p-1)^2}}\ .$$

The sign of $\left(f_p^{(\text{off})}\right)'$ is the sign of $g_p^{(\text{off})}$, where:

$$g_p^{(\text{off})}(\beta) = \beta^4(-9p^3+30p^2-27p+6)+\beta^3(-6p^2+18p-12)+\beta^2(-78p^2+126p+33)+\beta(18p-180)+81.$$

Differentiating again two times, we obtain:

$$\left(g_p^{(\text{off})}\right)'(\beta) = 4\beta^3(-9p^3+30p^2-27P+6)+3\beta^2(-6p^2+18p-12)+2\beta(-78p^2+126p+33)+18p-180\ ;$$

$\left(g_p^{(\text{off})}\right)''(\beta) = 12\beta^2(-9p^3+30p^2-27p+6)+6\beta(-6p^2+18p-12)-156p^2+252p+66.$

If $p \geq 4$,

$\left(g_p^{(\text{off})}\right)''(\beta) \leq 12\beta^2((-36+30)p^2-108+6)+6\beta((-24+18)p-12)+(-588+252)p+(-80+66) \leq 0.$

Now $\left(g_2^{(\text{off})}\right)''(\beta) = -54$ *and* $\left(g_3^{(\text{off})}\right)''(\beta) = -576\beta^2 - 72\beta - 582 \leq 0$, *thus for all* $p > 1$ *and* $1/p \leq \beta \leq 1$, $\left(g_p^{(\text{off})}\right)''(\beta) \leq 0$. *Therefore* $g_p^{(\text{off})}$ *is concave.*

We have $g_p^{(\text{off})}(1) = -9p^3 - 54p^2 + 135p - 72$, *thus for* $p \geq 3$, $g_p^{(\text{off})}(1) \leq p(-27-162+135) - 72 \leq 0$, *and* $g_2^{(\text{off})}(1) = -72 - 216 + 270 - 72 \leq 0$. $g_p^{(\text{off})}(1/p) \geq 21 - 35 + 15 \geq 0$. *We conclude that for all* $p > 1$, $f_p^{(\text{off})}$ *has a single maximum in* $[1/p, 1]$, *reached for* $\beta = \beta_p^{(\text{off})}$, *where* $g_p^{(\text{off})}(\beta_p^{(\text{off})}) = 0$. *Finally* OffLine-Greedy *is a* $f_p^{(\text{off})}(\beta_p^{(\text{off})})$ *approximation.*

Proposition 4 *The approximation factor* $f_p^{(\text{on})}(\beta_p^{(\text{on})})$ *for* OnLine-Greedy *cannot be improved.*

Proof 4 *Consider an instance with* p *processors and* $n = p(p-1)+1$ *jobs, where for all* $i \in \{1, \ldots, n-1\}$, $a_i = 1$, *and* $a_n = B = \frac{\beta_p^{(\text{on})} p(p-1)}{1-\beta_p^{(\text{on})}}$.

OnLine-Greedy *assigns* $p-1$ *unit-size jobs to each processor, and then the big job is assigned to any processor, leading to a power dissipation of:*

$$P_{\text{online}} = \left(\frac{S+(p-1)a_j}{p}\right)^3 + (p-1)\left(\frac{S-a_j}{p}\right)^3,$$

where $j = n$.

From $\beta_p^{(\text{on})} \geq 1/p$, *we deduce that* $B \geq p$. *Therefore the optimal solution assigns* J_n *to the first processor, and* p *unit-size jobs to each other processor. We have* $a_j = O = B$ *and for* $q \in \{2, \ldots, p\}$, $\sum_{i \in load(q)} a_i = p = (S-O)/(p-1)$, *hence*

$$P_{\text{opt}} = O^3 + (p-1)\left(\frac{S-O}{p-1}\right)^3 .$$

Moreover we have $O = \beta_p^{(\text{on})} S$:

$$O - \beta_p^{(\text{on})} S = B - \beta_p^{(\text{on})}(B + p(p-1)) = B - p(p-1)\beta_p^{(\text{on})}(\beta_p^{(\text{on})}/(1-\beta_p^{(\text{on})}) + 1) = 0 .$$

We conclude that, for this instance,

$$\frac{P_{\text{online}}}{P_{\text{opt}}} = f_p^{(\text{on})}(\beta_p^{(\text{on})}) .$$

Proposition 5 *The approximation factor* $f_p^{(\text{off})}(\beta_p^{(\text{off})})$ *for the ratio of the* OffLine-Greedy *cannot be improved.*

Proof 5 *Consider an instance with* p *processors and* $n = 2p+1$ *jobs, where for all* $i \in \{1, \ldots, p\}$, $a_{2i-1} = a_{2i} = 2p - i + v_i$, *and* $a_{2p+1} = p + v_p$, *and where* $a_{2p+1} = p + v_p$. *We define* A *as* $A = \frac{3p(1-\beta_p^{(\text{off})} p)}{\beta_p^{(\text{off})}(p+1)-3}$, *and* $v_i = \frac{i-1}{p-1}A$ *for all* $i \in \{1, \ldots, p\}$.

We first show that the jobs are sorted in non-increasing order:

- *For all* $i \in \{1, \ldots, p\}$, $a_{2i-1} = a_{2i}$
- $a_n = a_{n-1}$
- *For all* $i \in \{1, \ldots, p-1\}$, $a_{2i+1} - a_{2i} = -1 + v_{i+1} - v_i = -1 + \frac{A}{p-1}$. *Consider the function* $\hat{h}_p(\beta) \mapsto \frac{3p(\beta p-1)}{3-\beta(p+1)}$. *Its derivative is nonnegative, hence* $\hat{h}_p$ *is increasing. We show now that* $\beta_p^{(\text{off})} \leq 3/(2p+1)$, *which will ensure that* $A = \hat{h}_p(\beta_p^{(\text{off})}) \leq \hat{h}_p(3/(2p+1)) = p-1$. *In turn, this will ensure that* $v_i \leq 1$ *for all* i, *hence the desired inequality* $a_{2i+1} \leq a_{2i}$. *To show that* $\beta_p^{(\text{off})} \leq 3/(2p+1)$, *we prove* $g_p^{(\text{off})}(3/(2p+1)) \leq 0$. *Indeed we have* $g_p^{(\text{off})}(3/(2p+1)) = -135p(8p^3+3p^2-30p+19)/(2p+1)^4$, *and* $8p^3+3p^2-30p+19 \geq p(32-30)+19 \geq 0$. *Finally, since* $g_p^{(\text{off})}(1/p)$ *is positive,* $g_p^{(\text{off})}(3/(2p+1))$ *negative, we have* $\beta_p^{(\text{off})} \geq \frac{3}{2p+1} = \frac{6p-3}{4p^2-1}$, *hence the result.*

Before the assignment of the last job, all processor loads are perfectly balanced. OffLine-Greedy *first assigns* $J_1, J_2, \ldots, J_p$ *to* $\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_p$ *respectively. Then the greedy assigns* $J_{p+1}, J_{p+2}, \ldots, J_{2p}$ *to* $\mathcal{P}_p, \mathcal{P}_{p-1}, \ldots, \mathcal{P}_1$ *respectively. After these assignments, for every* $i \in \{1, \ldots, \lceil p/2 \rceil\}$, *the load of processor* $\mathcal{P}_{2i-1}$ *is* $a_{2i-1}+a_{2(p-(i-1))} = 3p+v_i+v_{p-(i-1)} = 3p+\frac{i-1}{p-1}A+\frac{p-i}{p-1}A = 3p+A$. *And for all* $i \in \{1, \ldots, \lfloor p/2 \rfloor\}$, *the load of processor* $\mathcal{P}_{2i}$ *is* $a_{2i} + a_{2(p-(i-1))-1} = a_{2i-1} + a_{2(p-(i-1))} = 3p + A$. *The last job* J_n *is assigned to any processor, and the power dissipated by* OffLine-Greedy *is:*

$$P_{\text{offline}} = \left(\frac{S + (p-1)a_j}{p}\right)^3 + (p-1)\left(\frac{S - a_j}{p}\right)^3 ,$$

where $j = n$.

The optimal solution assigns $J_1, J_2, \ldots, J_{p-1}$ *to* $\mathcal{P}_2, \mathcal{P}_3, \ldots, \mathcal{P}_p$ *respectively. Then it assigns* $J_p, J_{p+1}, \ldots, J_{2p-2}$ *to* $\mathcal{P}_p, \mathcal{P}_{p-1}, \ldots, \mathcal{P}_1$ *respectively. The last three jobs* J_{2p-1}, J_{2p} *and* J_{2p+1} *are assigned to* $\mathcal{P}_1$, *which is the most loaded processor.*

The loads of processors $\mathcal{P}_2, \mathcal{P}_3, \ldots, \mathcal{P}_p$ *are perfectly balanced in the optimal assignment. For all* $i \in \{1, \ldots, \lfloor p/2 \rfloor\}$, *the load of processor* $\mathcal{P}_{2i}$ *is* $a_{2i-1} + a_{2(p-i)} = 3p + v_i + v_{p-i)} = 3p + \frac{i-1}{p-1}A + \frac{p-i+1}{p-1}A = 3p + pA/(p-1)$. *For all* $i \in \{1, \ldots, \lceil p/2 \rceil - 1\}$, *the load of processor* $\mathcal{P}_{2i+1}$ *is* $a_{2i} + a_{2(p-i)-1} = a_{2i-1} + a_{2(p-i)} = 3p + pA/(p-1)$. *Then, the load of processor* $\mathcal{P}_1$ *is* $O = 3a_n = 3p + 3A \geq \sum_{i \in load(q)} a_i$ *for* $q \in \{2, \ldots, p\}$. *This implies that*

$$P_{\text{opt}} = O^3 + (p-1)\left(\frac{S-O}{p-1}\right)^3 .$$

There remains to show that we have $O = \beta_p^{(\text{off})} S$. *But*

$$S = 3p^2 + v_p + 2\sum_{i=1}^{p} v_i = 3p^2 + A + \frac{2A}{p-1}\sum_{i=0}^{p-1} i = 3p^2 + (p+1)A ,$$

hence

$$\beta_p^{(\text{off})} S - O = 3p(\beta_p^{(\text{off})} p - 1) + (\beta_p^{(\text{off})}(p+1) - 3)A = 3p(\beta_p^{(\text{off})} p - 1) + 3p(\beta_p^{(\text{off})} p - 1) = 0.$$

Finally, since $a_j = a_n = O/3$, *the ratio of this instance is*

$$\frac{P_{\text{offline}}}{P_{\text{opt}}} = f_p^{(\text{off})}(\beta_p^{(\text{off})}) .$$

	OnLine-Greedy		OffLine-Greedy	
p	$\beta_p^{(\mathrm{on})}$	$f_p^{(\mathrm{on})}(\beta_p^{(\mathrm{on})})$	$\beta_p^{(\mathrm{off})}$	$f_p^{(\mathrm{off})}(\beta_p^{(\mathrm{off})})$
2	0.577	1.866	0.513	1.086
3	0.444	2.008	0.350	1.081
4	0.372	2.021	0.267	1.070
5	0.325	2.001	0.216	1.061
6	0.292	1.973	0.181	1.054
7	0.266	1.943	0.156	1.048
8	0.246	1.915	0.137	1.043
64	0.0696	1.461	0.0177	1.006
512	0.0186	1.217	0.00223	1.00083
2048	0.00479	1.104	0.000278	1.00010
2^{24}	0.0000192	1.006	0.0000000680	1.000000025

Table 1: Numerical values for the approximation factors of OnLine-Greedy and OffLine-Greedy

2.2 The approximation factor as a function of p

We provide a few observations on the values of the approximation factor of OnLine-Greedy and OffLine-Greedy for large values of p. Using Taylor expansions, we derive the following asymptotic values for large p:

- For large p, $\beta_p^{(\mathrm{on})} = \left(\frac{2}{p^2}\right)^{1/3} + O(1/p)$. Note that $\sqrt[3]{2} \approx 1.260$.
- For large p, $\beta_p^{(\mathrm{off})} = \frac{3(1+\sqrt{79})}{26p} + O(1/p^2)$. Note that $\frac{3(1+\sqrt{79})}{26} \approx 1.141$.

It is worth pointing out that both OnLine-Greedy and OffLine-Greedy are asymptotically optimal when p is large, while in the case of makespan minimization, the asymptotic approximation factor of OnLine-Greedy was equal to 2 and that of OffLine-Greedy equal to 4/3.

For $p = 2$ we have exact values: $\beta_2^{(\mathrm{on})} = \frac{\sqrt{3}}{3}$ and $f_2^{(\mathrm{on})}(\beta_2^{(\mathrm{on})}) = 1 + \frac{\sqrt{3}}{2}$ while $\beta_2^{(\mathrm{off})} = \frac{\sqrt{91}-8}{3}$ and $f_2^{(\mathrm{off})}(\beta_2^{(\mathrm{off})}) = 1 + \frac{\sqrt{91}+10}{18} \approx 1.086$. We report representative numerical values in Table 1. We observe that OnLine-Greedy is at most 50% more costly than the optimal for $p \geq 64$, while OffLine-Greedy always remains within 10% of the optimal, and gets within 5% for $p \geq 7$.

3 Conclusion

In this short communication, we have fully characterized the performance of the greedy algorithm for the power minimization problem. On the practical side, further work could be devoted to conducting experiments with a more complicated power model, that would include static power in addition to dynamic power. With such a model, the "natural" greedy algorithm would assign the next job to the processor that minimizes the increment in total power. There would then be two choices, either the currently least loaded processor, or a currently unused processor (at the price of more static energy to be paid).

Acknowledgments. The authors are with Université de Lyon, France. A. Benoit and Y. Robert are with the Institut Universitaire de France. This work was supported in part by the ANR *StochaGrid* project.

References

[1] H. Aydin and Q. Yang. Energy-aware partitioning for multiprocessor real-time systems. In *Proceedings of the International Parallel and Distributed Processing Symposium (IPDPS)*, pages 113–121. IEEE CS Press, 2003.

[2] A. P. Chandrakasan and A. Sinha. Jouletrack: A web based tool for software energy profiling. In *Design Automation Conference*, pages 220–225, Los Alamitos, CA, USA, 2001. IEEE Computer Society Press.

[3] J.-J. Chen and T.-W. Kuo. Multiprocessor energy-efficient scheduling for real-time tasks. In *Proceedings of International Conference on Parallel Processing (ICPP)*, pages 13–20. IEEE CS Press, 2005.

[4] R. L. Graham. Bounds on multiprocessing timing anomalies. *SIAM Journal on Applied Mathematics*, 17:416–429, 1969.

[5] T. Ishihara and H. Yasuura. Voltage scheduling problem for dynamically variable voltage processors. In *Proceedings of International Symposium on Low Power Electronics and Design (ISLPED)*, pages 197–202, New York, NY, USA, 1998. ACM Press.

[6] P. Langen and B. Juurlink. Leakage-aware multiprocessor scheduling. *Journal of Signal Processing Systems*, 57(1):73–88, 2009.

[7] M. P. Mills. The internet begins with coal. *Environment and Climate News*, page ., 1999.

[8] R. Mishra, N. Rastogi, D. Zhu, D. Mossé, and R. Melhem. Energy aware scheduling for distributed real-time systems. In *Proceedings of the International Parallel and Distributed Processing Symposium (IPDPS)*, pages 21–29, 2003.

[9] K. Pruhs, R. van Stee, and P. Uthaisombut. Speed scaling of tasks with precedence constraints. *Theory of Computing Systems*, 43:67–80, 2008.