

Um Algoritmo de Escalonamento para Redução do Consumo de Energia em Computação em Nuvem

Pedro Paulo Vezz  Campos
Orientador: Prof. Dr. Daniel Mac do Batista

Departamento de Ci ncia da Computa  o — Instituto de Matem tica e Estat stica — Universidade de S o Paulo

Introdu  o

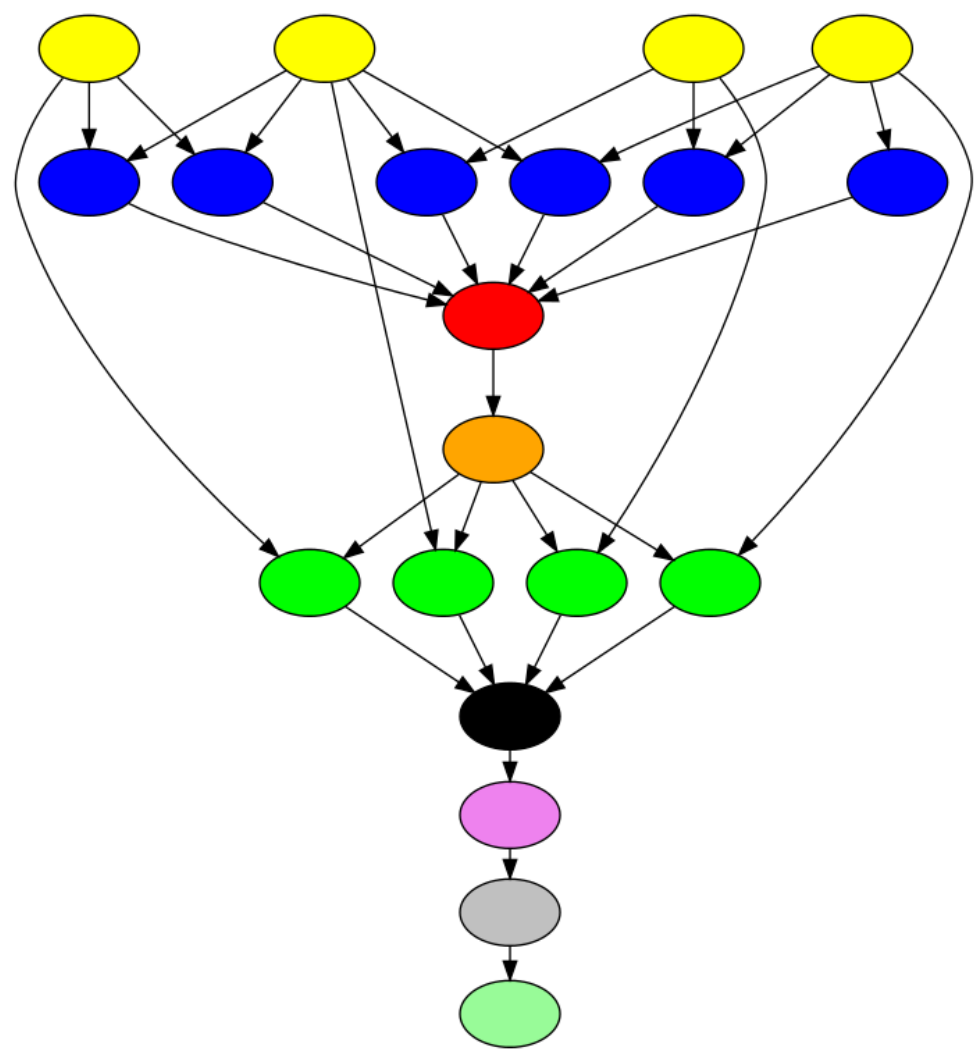
A **Lei de Moore**, que profetiza que o poder computacional de dispositivos dobra a cada 18 meses, est  chegando ao fim da sua vida [PH12]. Processadores atuais atingiram uma barreira de pot ncia mas no entanto n o eram eficientes no **consumo energ tico** [BH07]. Assim, novas tend ncias surgiram na ind stria: processadores mais simples, mais paralelos e mais eficientes.

Computa  o em nuvem surgiu como uma consequ ncia quase natural destas tend ncias. Ao consolidar poder de processamento, transfer ncia de dados e armazenamento   poss vel reduzir custos e desperd cios. Algumas estrat gias poss veis: **consolida  o** de m quinas virtuais, dimensionamento de tens o e frequ ncia (**DVFS**) e **algoritmos energeticamente eficientes**.

O desenvolvimento dos estudos contou com a contribui  o da aluna de mestrado **Elaine Naomi Watanabe**. Como resultado, este TCC apresenta um **novo algoritmo** de escalonamento de fluxos de trabalho em computa  o em nuvem voltado para a efici ncia energ tica. O desempenho foi comparado com o trabalho “*Energy-aware simulation with DVFS*” [GMDC+13] e com um algoritmo de escalonamento cl ssico mas sem um foco na efici ncia energ tica.

Modelagem

Podemos modelar algum processamento paralelo a ser computado como um **digrafo ac clico** (DAG). Um exemplo,   a aplica  o Montage da figura ao lado, que produz mosaicos astron micos. D vida: como associar uma tarefa a uma m quina de forma a diminuir o tempo de processamento ou energia consumida? O problema de achar um **escalonamento**  timo   **NP-dif cil**!



Algoritmo Proposto

ESCALONARPOWERHEFT(*tarefa*, *VM*)

- 1 *F* = filhos diretos da *tarefa* no DAG
- 2 Escalone *tarefa* em *VM*
- 3 Escalone *F* utilizando o algoritmo HEFT
- 4 *energia* = ESTIMARENERGIACONSUMIDA()
- 5 Volte para o escalonamento do come o do la o
- 6 **retorne** *energia*

POWERHEFTLOOKAHEAD()

- 1 *V* = {*VmMaisR pida*} // VMs usadas ao escalonar
- 2 *O* = os tipos de VMs que podem ser instanciadas
- 3 Ordene o conjunto de tarefas segundo o crit rio *rank_u*
- 4 **enquanto** h  tarefas n o escalonadas
- 5 *t* = a tarefa n o escalonada de maior *rank_u*
- 6 // Vamos tentar escalonar t em uma VM existente
- 7 **para** cada *v* em *V*:
- 8 ESCALONARPOWERHEFT(*t*, *v*)
- 9 // Vamos tentar escalonar t em uma nova VM
- 10 **para** cada *o* em *O*:
- 11 *V* = *V* ∪ {*o*}
- 12 Atualize os valores de *rank_u*
- 13 *t* = a tarefa n o escalonada de maior *rank_u*
- 14 ESCALONARPOWERHEFT(*t*, *o*)
- 15 Escalone *t* na VM que minimiza a energia consumida
- 16 Atualize *V* e *rank_u* caso necess rio

Agradecimentos

O autor gostaria de agradecer os valiosos coment rios fornecidos pelo orientador e por Elaine Watanabe, sem os quais este trabalho n o teria chegado a este ponto.

Escalonamento de fluxos de trabalho com computa  o em nuvem

O *Heterogeneous Earliest Finish Time* (**HEFT**) [THW02]   uma boa heur stica para o problema de escalonamento. Ele recebe como par metros um DAG a ser escalonado, um conjunto possivelmente heterog neos de m quinas que realizar o o processamento, os tempos de processamento de cada tarefa em cada m quina e o tempo de transmiss o entre duas tarefas. Ele   dividido em duas fases: **prioriza  o** e **sele  o**.

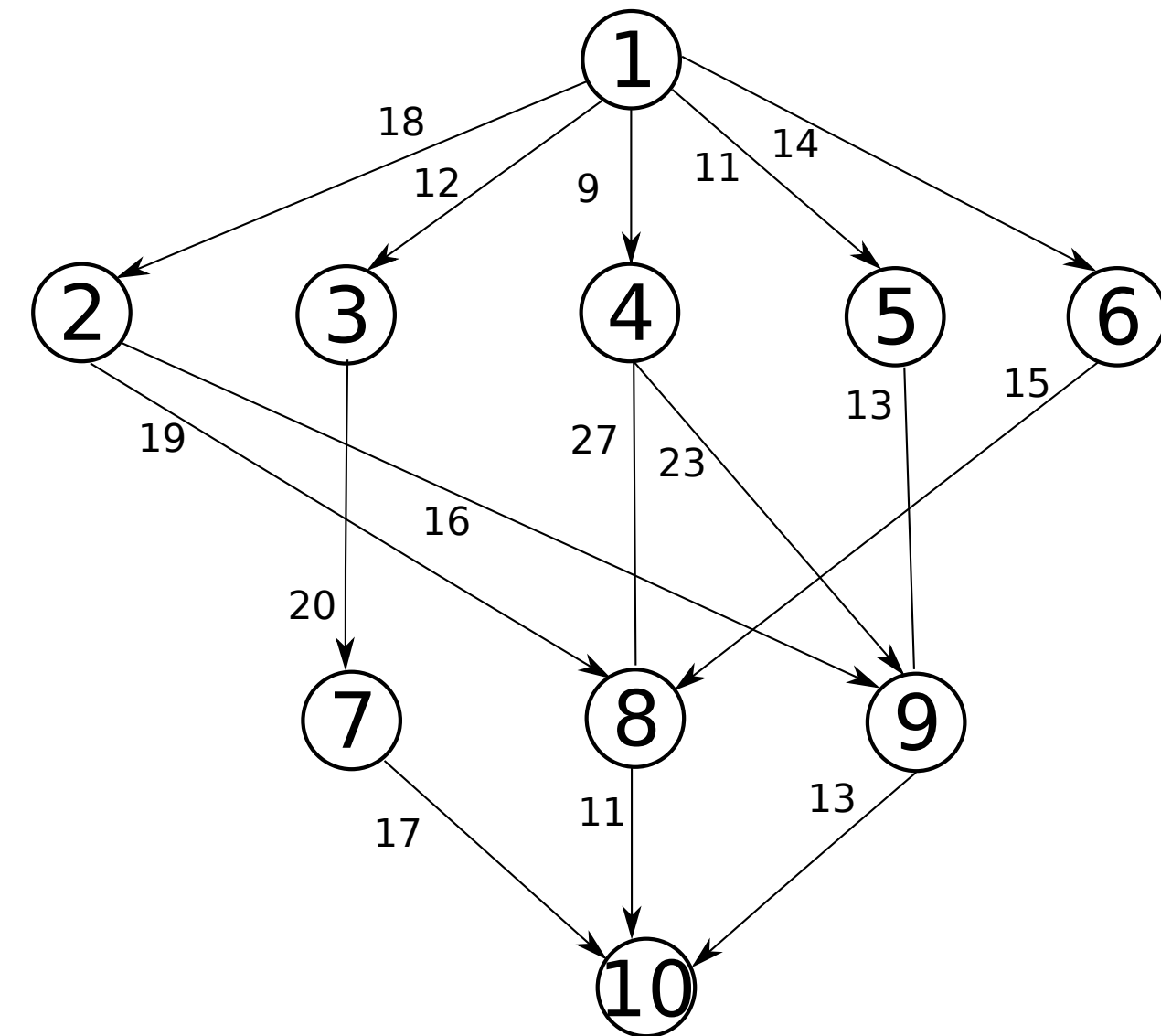
Prioriza  o

Qual tarefa escalonar primeiro? A f rmula abaixo   o crit rio de prioriza  o do HEFT. Al m de gerar uma **ordem topo-l gica**, d  prioridade a tarefas mais cr ticas do DAG.

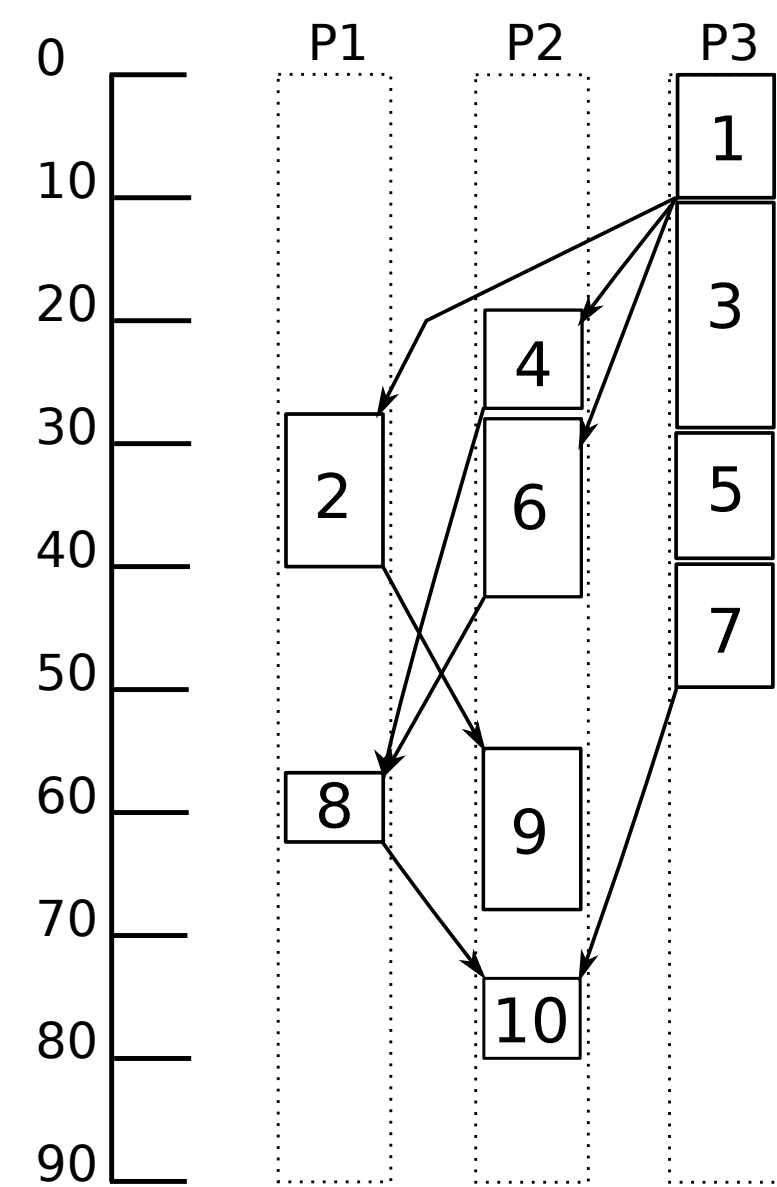
$$rank_u(n_i) = w_i + \max_{n_j \in succ(n_i)} (\bar{c}_{i,j} + rank_u(n_j))$$

Sele  o

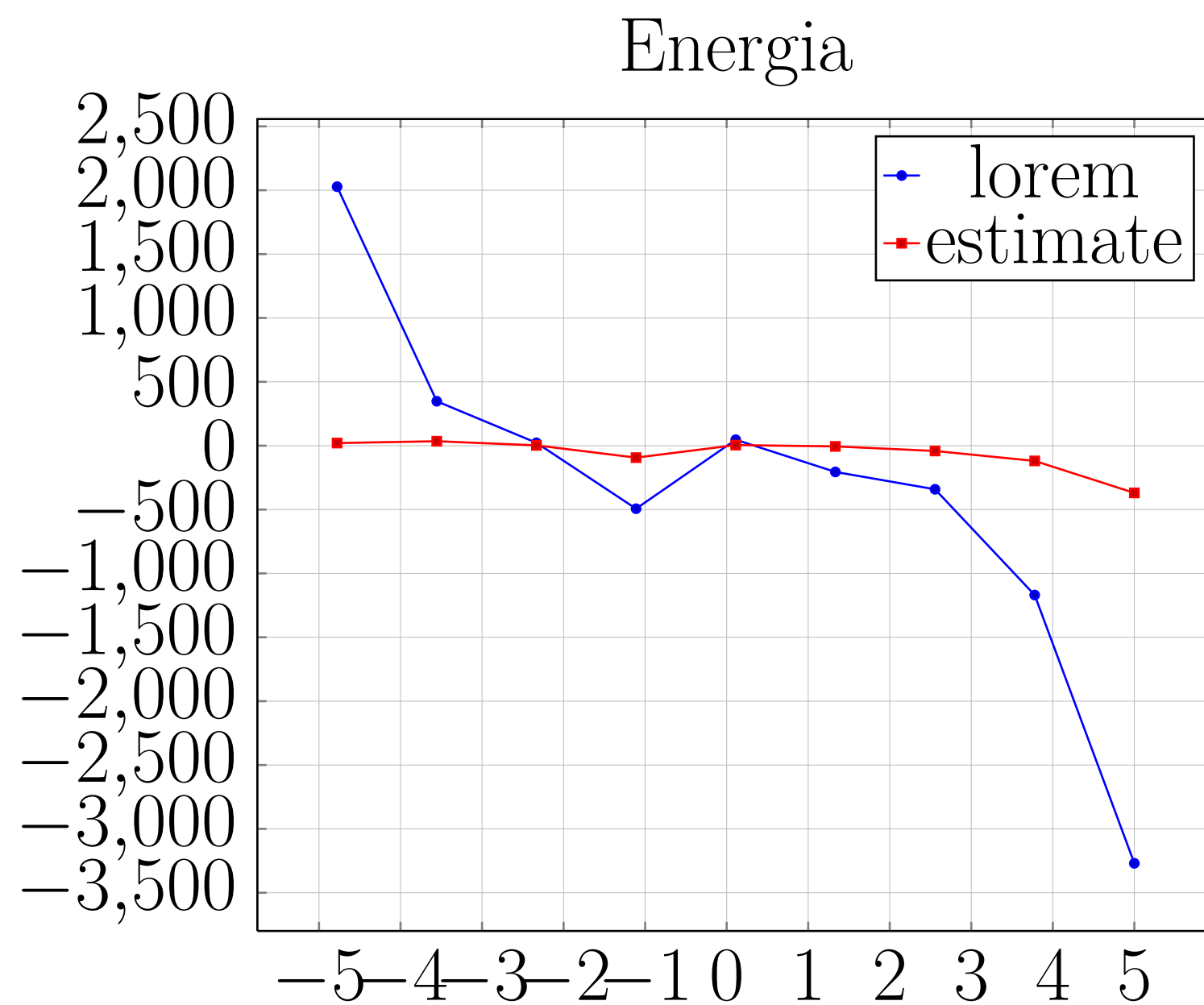
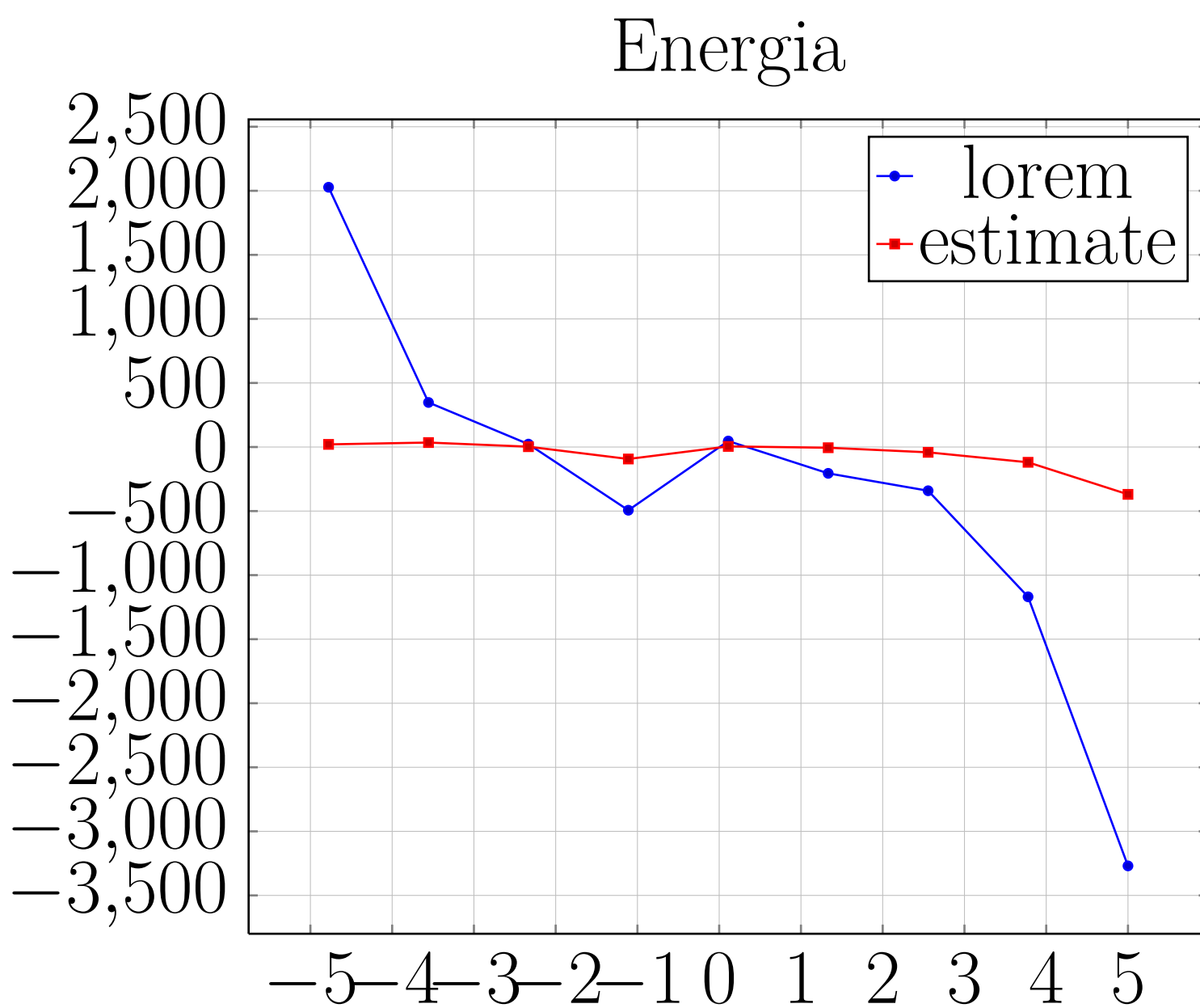
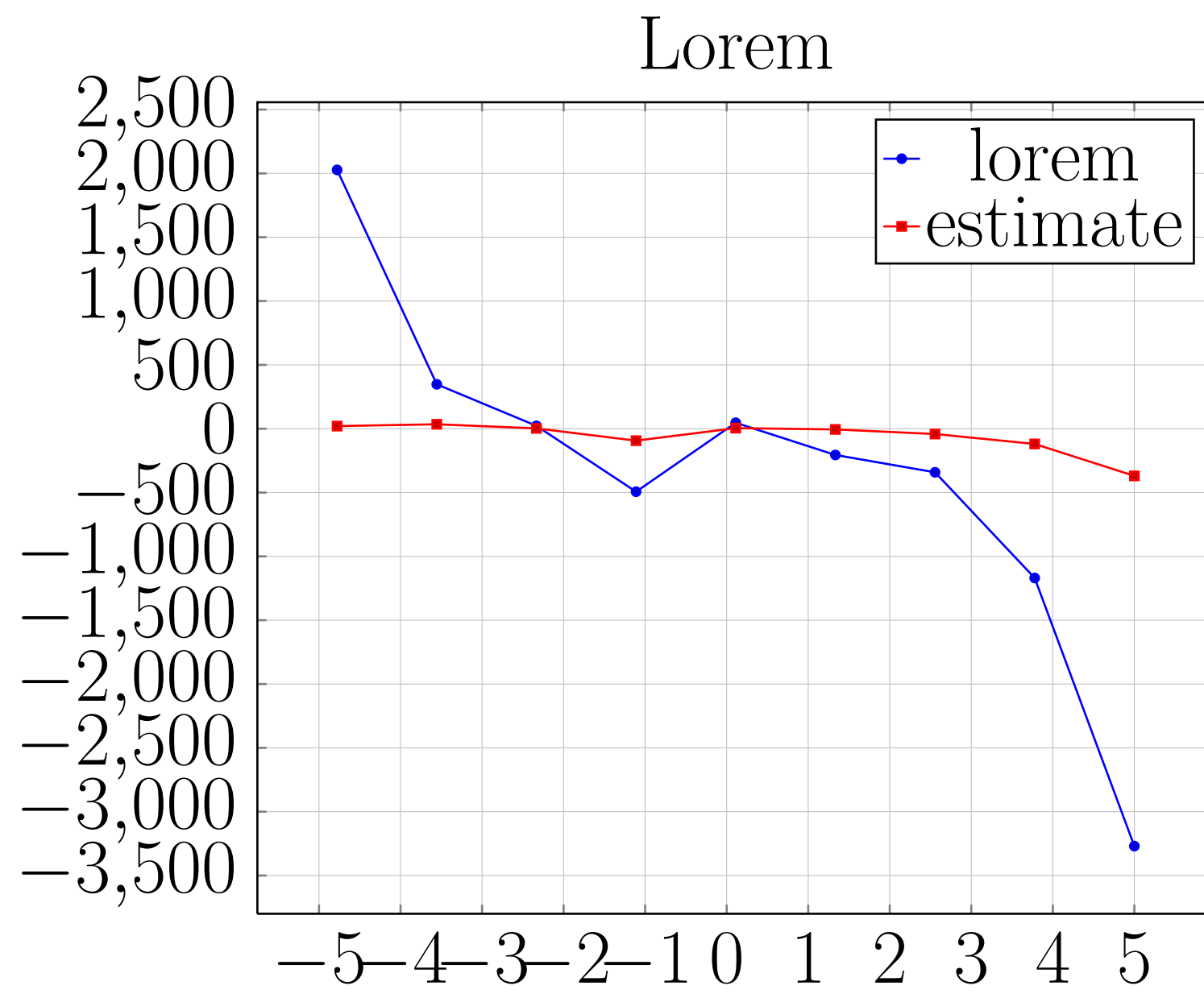
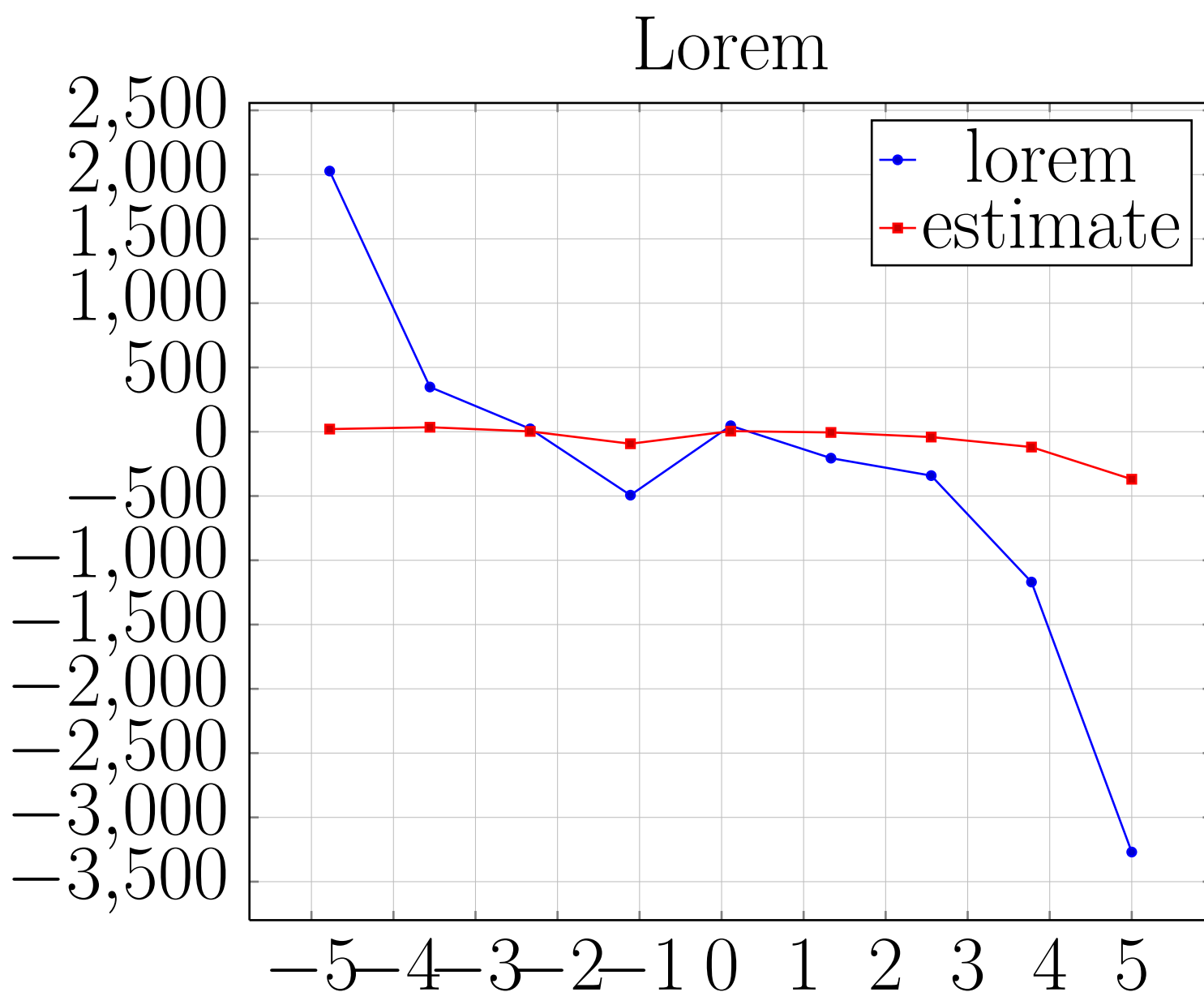
Em qual m quina escalonar uma tarefa? O HEFT tenta minimizar o **tempo mais cedo de conclus o** de cada tarefa na esperan a que isso minimize a conclus o do fluxo de trabalho como um todo.



Tarefa	P1 (s)	P2 (s)	P3 (s)	rank _u (n _i)
1	14	16	9	108.000
2	13	19	18	77.000
3	11	13	19	80.000
4	13	8	17	80.000
5	12	13	10	69.000
6	13	16	9	63.333
7	7	15	11	42.667
8	5	11	14	35.667
9	18	12	20	44.333
10	21	7	16	14.667



Resultados



Refer ncias

- [PH12] D.A. Patterson e J.L. Hennessy. *Computer Organization and Design: The Hardware/software Interface*. Morgan Kaufmann Series in Computer Graphics. Morgan Kaufmann, 2012.
- [BH07] L.A. Barroso e U. H lzl . The case for energy-proportional computing. *Computer*, 40(12):33-37, 2007.
- [GMDC+13] Tom Gu rout, Thierry Monteil, Georges Da Costa, Rodrigo Neves Calheiros, Rajkumar Buyya e Mihai Alexandru. Energy-aware simulation with dvfs. *Simulation Modelling Practice and Theory*, 2013.
- [THW02] H. Topcuoglu, S. Hariri e Min-You Wu. Performance-effective and low-complexity task scheduling for heterogeneous computing. *Parallel and Distributed Systems, IEEE Transactions on*, 13(3):260-274, 2002.