# Estratégia para economizar energia de nós ociosos de um cluster

**Fábio Diniz Rossi, César A. F. De Rose**

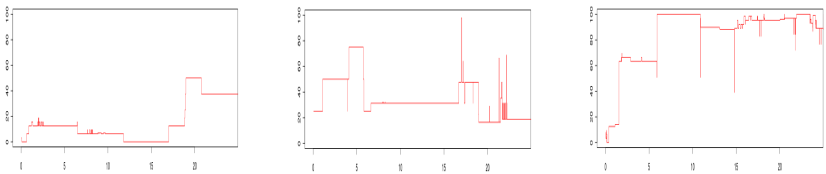
[1]Programa de Pós-Graduação em Ciência da Computação
Pontifícia Universidade Católica do Rio Grande do Sul
Av. Ipiranga, 6681 – Prédio 32 – Porto Alegre – RS – Brasil

fabio.diniz@acad.pucrs.br, derose@pucrs.br

***Resumo.*** *Atualmente, existe uma demanda na redução do consumo de energia em ambientes de HPC, devido tanto ao impacto sobre o custo financeiro, quanto ao impacto ambiental por meio da emissão de gases. Este trabalho propõe uma estratégia de gerenciamento de energia que desliga nodos ociosos do cluster.*

## 1. Motivação, proposta e resultados preliminares

A motivação deste trabalho consiste em avaliar a redução no consumo de energia [Maillard et al. 2010] em um cluster, quando nodos ociosos são desligados. Para tanto, foi criada uma estratégia e avaliada com a utilização do simulador SimGrid [Neves et al. 2012] com um módulo adicional para avaliar consumo de energia. O ambiente simulado apresenta um cluster com 128 nodos, com dois escalonadores distintos: FCFS (*First-Come, First-Served*) e CBF (*Conservative BackFilling*). Os escalonadores foram implementados em dois modos: **idle**, quando os nodos ociosos permanecem ligados; **poff** quando os nodos ociosos são desligados e religados quando de uma submissão. Como podemos ver na Figura 1 (eixo x: tempo, eixo y: número de nodos), utilizamos um *trace* real do Centro de Computação de San Diego[1]. Foram escolhidos três períodos de 24 horas cada, com comportamentos distintos dos *traces*: baixa utilização 1(a), média utilização 1(b) e alta utilização 1(c).

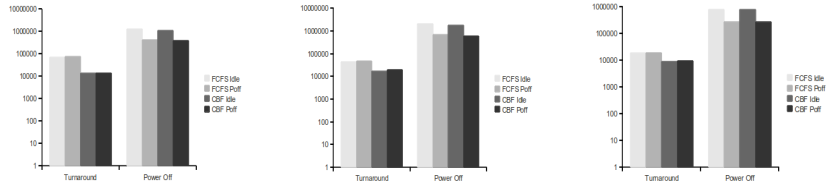


(a) Baixa utilização (b) Média utilização (c) Alta utilização

**Figura 1. Comportamento dos traces**

Em nossas avaliações, foram levadas em consideração duas métricas: *turnaround* e consumo de energia (*power off*). É importante que o ônus desta estratégia sobre o *turnaroud* seja o menor possível, pois o tempo de execução é um fator determinante nos ambientes de HPC. A Figura 2 (eixo x: métricas, eixo y: valores em tempo para turnaround e watts para consumo de energia) apresenta o resultado das simulações realizadas sobre cada *trace*, utilizando os dois escalonadores, nas duas implementações propostas. Os resultados apresentados para cada *trace* permitem comparação dos modos **idle** e **poff** em cada um dos escalonadores.

[1]http://www.cs.huji.ac.il/labs/parallel/workload/logs.html




(a) Baixa utilização (b) Média utilização (c) Alta utilização

**Figura 2. Turnaround e Power Off**

O resultado da simulação 2(a) para o escalonador FCFS, apresenta um aumento no *turnaround* de 7.4% e uma redução no consumo de energia de 65.8%. Na mesma simulação, escalonador CBS apresenta um aumento no *turnaround* de 2.9% e uma redução no consumo de energia de 64.2%. A simulação apresentada em 2(b), mostra o escalonador FCFS com um aumento de 11.5% no *turnaround* e uma redução no consumo de energia de 67.1%. Para o mesmo ambiente, o escalonador CBS mostra um aumento no *turnaround* de 16.6% e uma redução no consumo de energia de 66.3%. No último ambiente simulado, apresentado em 2(c), podemos ver que o escalonador FCFS aponta aumento no *turnaround* de 5.4% e uma redução no consumo de 66.1%. Já o escalonador CBS, indica um aumento no *turnaround* de 6.6% e uma redução no consumo de energia de 66.1%. Embora os resultados mostrem uma grande redução no consumo de energia, o aumento do *turnaround* é indesejado. Este aumento não impacta diretamente às características impostas pelo escalonador FCFS, mas atrasa a execução de todos os *jobs* submetidos. Quanto ao escalonador CBF, que tem como característica garantir o tempo de início de execução de cada *job*, esta estratégica influencia diretamente, devido ao CBF não levar em consideração os tempos entre desligar e religar os nodos do cluster para a realização do cálculo do início da execução dos *jobs*.

## 2. Conclusões e Perspectivas

Este artigo mostrou um trabalho preliminar que visa reduzir o consumo de energia em um cluster, tentando ao máximo, manter o *turnaround*. A abordagem apresentada desliga os nodos ociosos quando uma tarefa finaliza, sem verificar se existem tarefas alocadas na fila do gerenciador de recursos. Esta abordagem se mostra útil em cenários em que o comportamento dos *jobs* submetidos não utiliza a totalidade do cluster mantendo nodos ociosos. Portanto, podem existir casos em que manter os nodos ligados para suportar uma nova tarefa pode ser mais vantajoso, do que desligá-los e religá-los. Assim, será implementada uma nova abordagem que verifica se existem tarefas na fila do gerenciador de recursos, a quantidade de nodos que estão alocados para esta tarefa e qual o tempo de alocação. Com estas informações, esta nova abordagem poderá decidir se nodos ociosos pelo fim de alguma tarefa, devam permanecer ligados, ou desligá-los. Outra oportunidade consiste em não desligar os nodos ociosos, do contrário, mantê-los em estado de hibernação ou *standby*. Esta abordagem possibilitaria um retorno do sistema operacional mais rápido do que o estado de *power off*.

## Referências

Maillard, N., Navaux, P., and De Rose, C. (2010). Energy-aware scheduling of parallel programs. In *Conferencia Latino Americana de Computacion de Alto Rendimiento*, CLCAR, pages 95–101.

Neves, M. V., Ferreto, T., and Rose, C. (2012). Scheduling MapReduce Jobs in HPC Clusters. In *Euro-Par 2012 Parallel Processing*, pages 179–190, Berlin, Heidelberg. Springer Berlin Heidelberg.