Análise de soluções para *Grid Computing*

JONES BUNILHA RADTKE[1]; HENRIQUE DE VASCONCELLOS RIPPEL[2]; GERSON PORCIÚNCULA SIQUEIRA[3]; CARLOS VINÍCIUS RASCH ALVES[4]

[1]*Faculdade de Tecnologia SENAC Pelotas (FATEC) – jones.radtke@gmail.com*
[2]*Faculdade de Tecnologia SENAC Pelotas (FATEC) – hvrippel@gmail.com*
[3]*Faculdade de Tecnologia SENAC Pelotas (FATEC) – gersonporciuncula@gmail.com*
[4]*Faculdade de Tecnologia SENAC Pelotas (FATEC) – cvalves@senacrs.edu.br*

1. INTRODUÇÃO

O crescimento do conhecimento científico está diretamente relacionado a evolução das tecnologias computacionais, visto que, cada vez mais são utilizados sistemas de alto desempenho computacional, como ferramenta, para a resolução de cálculos matemáticos complexos, criação de modelos de simulação e processamento de renderização de imagens e vídeos na produção cinematográfica. Ainda, pesquisas na área da Medicina, Biologia e das Ciências Exatas são altamente beneficiadas com o uso destes sistemas.

A aquisição de sistemas computacionais que tem como característica principal um alto poder de processamento e armazenamento, também chamados de supercomputadores, requerem um alto investimento financeiro. Em conjunto à ideia de que recursos computacionais ociosos possam ser mais bem aproveitados, o uso de sistemas de *Grid Computing* torna-se uma alternativa, onde por intermédio de um único ambiente, pesquisadores podem submeter, tarefas, *jobs* e aplicações, para serem processados em algumas centenas de computadores, que dispõe de menor recurso de *hardware*, mas que atuam em conjunto de forma paralela e distribuída geograficamente. *Grid Computing* são sistemas computacionais com uma plataforma heterogênea e distribuída geograficamente de forma dispersa, onde os recursos são compartilhados de forma transparente para seus usuários que a utilizam por intermédio de um único ambiente DANTAS(2005).

O objetivo do estudo é efetuar uma análise comparativa entres as soluções OAR OAR(2015) e HTCondor HTCONDOR(2015), que destinam-se a implantação de grades computacionais, apresentando os testes efetuados e seus resultados, a fim de, determinar o desempenho dos métodos de escalonamento utilizados por cada solução.

2. METODOLOGIA

Com o propósito de verificar o desempenho, a performance dos métodos de escalonamento, a eficiência e mensurar a utilização de recursos consumidos por cada solução, foi estruturado um ambiente virtualizado, com a ferramenta XenServer versão 6.5, para a instalação dos *grids*. Ambas as estruturas, OAR e HTCondor, foram compostas por um elemento de submissão de tarefas, um gerenciador e dez executores, com *hardware* de uma CPU de 4 *cores*, 512MB de memória RAM, totalizando 40 *cores* de processamento e 5,12GB de memória RAM de recursos disponíveis em cada estrutura. Com o objetivo de comparação dos resultados, testes realizados no ambiente desenvolvido para esse estudo, também foram efetuados na estrutura do Grid5000 GRID5000 (2015), na França, utilizando dois *frontends* do *grid,* Grenoble, composto por 234 *cores*, divididos em três (3) *clusters* e o *frontend* Lyon, composto por 106 *cores*, divididos em quatro (4) *clusters*.

Os testes consistem na submissão de algoritmo para o *grid*, com a utilização da API própria de cada solução, por intermédio do elemento *frontend* (OAR) ou *submit* (HTCondor), gerando carga de processamento, uso de memória e recursos de redes em cada elemento que compõe a estrutura, desta forma, verifica-se o tempo de execução de cada tarefa. Para a execução de algoritmos com essa finalidade, é necessário que sejam desenvolvidos baseados em Sistemas Paralelos e que possuam métodos de escalonamento em seu código. Utilizou-se algoritmos de divisão de carga, síntese de códigos irregulares e otimização de memória hierárquica, disponível no projeto Galois, GALOIS(2015). O foco principal desta análise é mensurar a eficiência com relação aos recursos utilizados de cada solução, com tudo, não serão abordados os mecanismos de desenvolvimento do código utilizado. Os recursos de cada solução foram mensurados através da ferramenta de monitoramento Ganglia, GANGLIA(2015), que é um sistema desenvolvido para ser utilizado em ambientes distribuídos e de alto desempenho. O primeiro teste consiste na submissão do algoritmo para cada solução com sua estrutura de forma completa, com todos os elementos executores ativos. No teste seguinte haverá uma variação da quantidade de elementos executores ativos, em cada estrutura, e também no número de tarefas, *job*, submetidos, desta forma, é avaliado o desempenho do escalonador de cada *grid* nestas condições. O terceiro teste tem por objetivo verificar os resultados na submissão de tarefas utilizando-se da funcionalidade de recursos "desejados" disponível na solução OAR.

3. RESULTADOS E DISCUSSÃO

A Figura 1 é apresentada um gráfico com os tempos de execução do algoritmo em cada solução. Observa-se que o HTCondor foi a solução com melhor desempenho, seguido pelo OAR e o Grid5000 para o teste de primeira execução. Ainda neste gráfico é possível verificar o quanto é eficiente um sistema de *grid*, em comparação a execução do mesmo algoritmo em um computador, *localhost*, que possui os mesmos recursos de *hardware* de um elemento executor do *grid*.

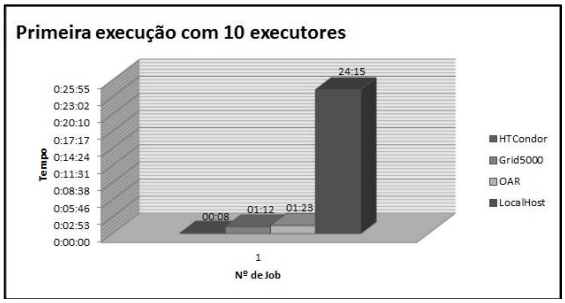

Figura 1 - Gráfico de primeira execução do algoritmo

Os tempos de execução, nos testes seguintes, serão relativamente menores, pois o *grid* utiliza recursos de otimização de memória, presente no código do algoritmo utilizado, aumentando a eficiência de execução das tarefas. Tal recurso do algoritmo provê paralelizar as tarefas com utilização de memória compartilhada dentro da estrutura do *grid*. Os resultados dos testes apresentados nos gráfico da Figura 2, foram obtidos por meio da submissão de um conjunto de tarefas simultâneas, dividas em 50, 100, 500 e 2000 tarefas, para quatro ambientes com configurações distintas, onde a primeira foi composta por um executor, o segundo por três executores, o terceiro com cinco executores e o último com dez executores. A submissão de 50 tarefas também foi efetuada nos dois *frontend* do

Grid5000. Devido a uma política existe neste *grid*, o máximo de tarefas submetidas por único usuário não deve ultrapassar o valor de 50. O propósito desta sequencia de testes é avaliar o desempenho e a eficiência dos métodos de escalonamento de cada solução e o gerenciamento de recursos, quando há um volume de tarefas em fila de execução.

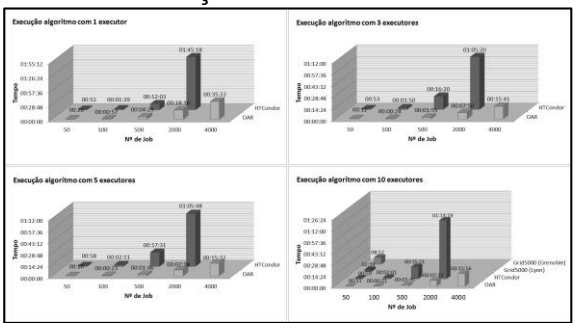

Figura 2 - Gráfico de execução do algoritmo em função da quantidade de tarefas submetidas

O OAR foi a solução mais eficiente, pois executou em menor tempo todas as tarefas, em todos cenários testados, toda via, é a solução que utilizou mais recursos de *hardware*, conforme os gráficos da Figura 3, apresentam a utilização dos recursos de CPU, memória RAM e tráfego de rede de cada solução em função do tempo, na execução de 2000 tarefas submetidas a cada *grid*. Analisando-os, é possível constatar que no OAR, o elemento que possui maior carga de processamento é o *Server Node*, uma vez que é o servidor de banco de dados do sistema, responsável por armazenar todas as informações relacionadas à execução das tarefas e gerenciamento de recursos do *grid*. Já para o HTCondor, o elemento com maior carga de processamento é o *Submit Machine*, devido este ser o responsável por efetuar o agendamento das tarefas submetidas ao *grid*.

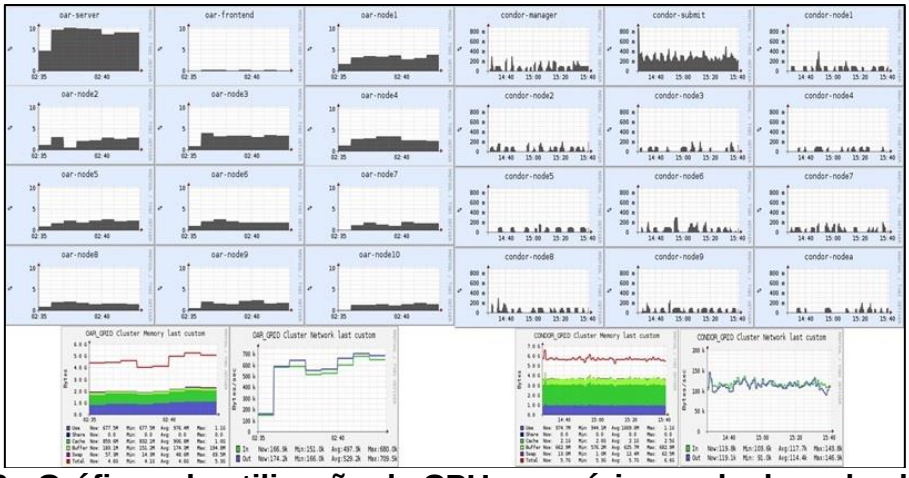

Figura 3 - Gráficos da utilização de CPU, memória e rede de cada elemento dos *grids* OAR e HTCondor com 2000 tarefas e 10 executores ativos.

O gráfico apresentado na Figura 4 exibe o resultado da submissão de 50 tarefas simultâneas com a seleção de recurso "desejado", funcionalidade presente no OAR, a ser executado em 12, 20 e 40 *cores* de processamento. Na estrutura desenvolvida para os testes, a cada 4 *cores*, representa um elemento executor, portanto, 12 *cores* condiz a 3 elementos executores. Observa-se que o melhor desempenho foi obtido pelo OAR comparado ao Grid5000, mas a principal constatação é que quanto maior o recurso "desejado", maior será o tempo de execução das tarefas concorrentes, pois o *grid* só iniciará a execução de uma

tarefa, quando houver a disponibilidade do recurso "desejado", consequentemente, acrescendo de tempo para o início de execução de cada tarefa.

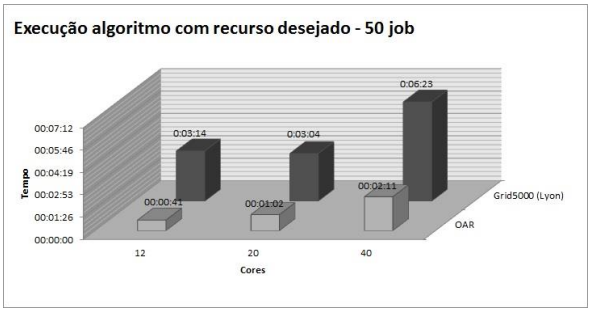

Figura 4 - Gráfico de execução do algoritmo com seleção de recurso desejado.

4. CONCLUSÕES

Conclui-se que ambas as soluções estudas, OAR e HTCondor, desempenham de forma satisfatório o objetivo proposto. Fazem utilização de recursos ociosos, na execução das tarefas submetidas ao sistema, sem a oneração e prejuízos aos elementos que compõe o \emph{grid}, indo de encontro com uma das premissas de um *Grid Computing*, portanto, são boas alternativas de implantação em ambientes que necessitam de sistemas com mais recursos computacionais. A diferença de eficiência comprovada, por meio dos resultados dos testes efetuados, é diretamente relacionada aos métodos de escalonamento e gerenciamentos dos recursos utilizados por cada *grid*. O HTCondor mostrou-se mais eficiente na execução de algoritmo em primeira instância. Por sua vez o OAR mostrou-se mais eficiente no gerenciamento da fila de execução de tarefas, e paralelismo das tarefas, igualmente, foi a solução que utilizou mais recursos de *hardware* dos elementos executores e DE rede.

Ainda conclui-se que o desempenho na execução de tarefas em um *grid*, está diretamente relacionado aos requisitos necessários para a execução desta tarefa. Se determinado algoritmo necessitar de mais recursos para ser executado, o *grid* utilizará dos métodos de gerenciamento de recursos para verificar quando haverá a viabilidade de sua execução, característica comprovada com a submissão de tarefas para o Grid5000, onde mesmo sendo uma estrutura que dispõe de mais recursos, apresentou maior tempo de execução, em virtude da concorrência pelos recursos disponíveis com as tarefas que já se encontravam aguardando na fila de execução do sistema.

5. REFERÊNCIAS BIBLIOGRÁFICAS

DANTAS, M.A.R. **Computação distribuída de alto desempenho: redes, clusters e grids computacionais**. Axcel books, 2005.

GALOIS. **Galois**. Acessado em mai. de 2015. Disponível em: http://iss.ices.utexas.edu/?p=projectst/.

GANGLIA. **Ganglia**. Acessado em abr. de 2015. Disponível em: http://ganglia.sourceforge.net/.

GRID5000. **Grid5000**. Acessado em mai. de 2015. Disponível em: https://www.grid5000.fr.

HTCONDOR. **Htcondor**. Acessado em mai. 2015. Disponível em: http://research.cs.wisc.edu/htcondor/.

OAR. **Oar**. Acessado em mar. de 2015. Disponível em: http://oar.imag.fr/.