Escalabilidade de Aplicações Bag-of-Tasks em Plataformas Heterogêneas

Jaime Freire de Souza[1], **Fabrício Alves Barbosa da Silva**[2], **Hermes Senger**[1]

[1] Departamento de Computação - Universidade Federal de São Carlos
Rod. Washington Luís, Km 235 - Caixa Postal 676 - São Carlos-SP

{jaime.souza,hermes}@ufscar.br

[2]Fundação Oswaldo Cruz - FIOCRUZ
Av. Brasil 4365, Rio de Janeiro, Brazil

fabricio.a.b.silva@gmail.com

Abstract. *Bag-of-Tasks (BoT) applications are parallel applications composed of independent (i.e., embarrassingly parallel) tasks, which do not communicate with each other, may depend upon one or more input files, and can be executed in any order. BoT applications are very frequent in several scientific areas, and it is the ideal application class for execution on large distributed computing systems composed of hundreds to many thousands of computational resources. This paper focusses on the scalability of BoT applications running on large heterogeneous distributed computing systems organized as a master-slave platform. The results demonstrate that heterogeneous master-slave platforms can achieve higher scalability than homogeneous platforms for the execution of BoT applications, when the computational power of individual nodes in the homogeneous platform is fixed. However, when individual nodes of the homogeneous platform can scale-up, experiments show that master-slave platforms can achieve near linear speedups.*

Resumo. *Aplicações Bag-of-Tasks (BoT) são aplicações paralelas compostas de tarefas independentes (ou seja, embaraçosamente paralelas), que não se comunicam entre si, podem depender de um ou mais arquivos de entrada e podem ser executadas em qualquer ordem. As aplicações BoT são muito frequentes em diversas áreas e comumente executadas em grandes sistemas de computação distribuída, como nas grades computacionais ou na nuvem. Este trabalho estuda a escalabilidade de aplicações BoT executando em grandes sistemas de computação distribuída heterogêneos organizados como uma plataforma mestre-escravo. Os resultados mostram que plataformas mestre-escravo heterogêneas podem alcançar limites de escalabilidade mais altos que as plataformas homogêneas para a execução de aplicações BoT, quando o poder computacional dos nós individuais da plataforma homogênea é fixo. No entanto, quando nós individuais da plataforma homogênea podem escalar verticalmente, é mostrado neste trabalho que plataformas homogêneas apresentam escalabilidade próxima do linear.*

1. Introdução

O presente trabalho apresenta um estudo sobre a execução de aplicações *Bag-of-tasks* (BoT) em plataformas computacionais distribuídas compostas por recursos heterogêneos.

Aplicações BoT representam uma parcela significativa das cargas de trabalho científicas executadas corriqueiramente sobre plataformas computacionais distribuídas, incluindo os portais científicos (*science-gateways*), *clusters*, grades computacionais e nuvem [14, 19]. Aplicações BoT são compostas por tarefas sequenciais e independentes, que não se comunicam entre si, tal como observado em aplicações embaraçosamente paralelas. A entrada para cada tarefa é composta de um ou mais arquivos, e um mesmo arquivo pode ser a entrada para mais de uma tarefa. Cada tarefa gera seu próprio conjunto de arquivos de saída, que podem ser compostos de um ou mais arquivos. Exemplos de aplicações BoT incluem simulações de Monte Carlo, buscas massivas (como quebra de chave), aplicações de manipulação de imagens e algoritmos de mineração de dados. Elas são frequentes em áreas como astronomia, física de alta energia, mineração de dados [26], bioinformática [26], e muitas outras.

A escalabilidade é uma propriedade relacionada a uma combinação algoritmo-máquina, em vez de ser uma propriedade exclusiva da arquitetura da máquina ou do algoritmo [27]. A análise de escalabilidade permite a identificação dos gargalos de desempenho do sistema, um melhor entendimento do efeito desses gargalos sobre a escalabilidade do sistema e também oferece subsídios para mitigar esses gargalos e promover a melhoria da escalabilidade.

Há vários estudos sobre a escalabilidade de aplicações BoT em plataformas de computação distribuída, como por exemplo em [11, 13, 12, 25, 13, 12]). Nesses estudos, as arquiteturas eram todas homogêneas, ou seja, eram compostas de recursos homogêneos. Contudo, verifica-se que a grande parte das aplicações de alto desempenho são atualmente executadas em sistemas de computação heterogêneos. Nos centros de computação de alto desempenho, é comum a presença de várias gerações de máquinas sendo utilizadas para a execução das cargas de trabalho corriqueiras. Plataformas distribuídas, incluindo a nuvem, federações de nuvem e grades computacionais disponibilizam recursos (em geral virtualizados) sob a forma de instâncias com variadas capacidades em termos de processamento, memória e armazenamento. Por exemplo, o inventário do Grid5000 relata ao menos 25 modelos diferentes de processadores e nove tipos de interconexão[1]. A nuvem tornou-se uma plataforma interessante para a execução de aplicações de HPC com custo total de propriedade reduzido, permitindo alocar rapidamente grandes quantidades de recursos e pagando pelo uso. Na computação em nuvem, a capacidade de computação é empacotada na forma de instâncias (máquinas virtuais, contêineres, ou máquinas dedicadas) e entregue na forma de um serviço de utilidade. É bastante comum provedores de recursos de nuvem oferecerem dezenas de instâncias com as mais variadas configurações e preços. A título de exemplo, um provedor disponibiliza mais de uma centena de configurações com diferentes capacidades[2].

Este artigo apresenta um estudo sobre a escalabilidade de aplicações BoT, quando estas são executadas em plataformas de computação distribuída que apresentam recursos computacionais heterogêneos. Mais precisamente, o presente estudo tem por objetivo investigar qual é o impacto do uso de nós de processamento com capacidades heterogêneas na execução de aplicações BoT, com respeito à escalabilidade. Como principais

[1]Informações coletadas de `https://www.grid5000.fr/mediawiki/index.php/Hardware` em novembro de 2018

[2]`https://aws.amazon.com/ec2/instance-types/?nc1=h\_ls`

contribuições, este artigo demonstra que plataformas mestre-escravo heterogêneas podem alcançar limites de escalabilidade melhores que sua contraparte homogênea para a execução de aplicações BoT, quando o poder computacional de nós individuais em plataformas homogêneas é fixo. Tais efeitos são estudados para aplicações com diferentes perfis. O restante do artigo é assim organizado. A próxima seção apresenta os modelos de aplicação e plataforma estudados. Os trabalhos relacionados são discutidos na seção 3. A seção 4 discute sobre a escalabilidade de sistemas. Os experimentos são apresentados na seção 5, juntamente com a discussão dos resultados. Por fim, as conclusões e trabalhos futuros são apresentados na seção 6.

2. Modelo de Aplicação e de Plataforma

Tipicamente, uma aplicação BoT A é composta por k tarefas independentes, em que a quantidade de computação associada a cada tarefa é pré-definida e pode variar entre as tarefas. No caso de tarefas homogêneas, cada tarefa executa uma quantidade de computação w_i, sendo que w_i é o mesmo valor para todas as tarefas. Já em tarefas heterogêneas, as quantidades de computação w_i e w_j podem ser diferentes, para tarefas t_i e t_j distintas. Em qualquer caso, a quantidade de computação w_i é fixa e não pode ser modificada arbitrariamente. Para executar, cada tarefa depende de um ou mais arquivos de entrada que podem ser compartilhados entre duas ou mais tarefas. Assume-se que a quantidade de dados associada a cada tarefa (tamanho dos arquivos de entrada) é fixo e não pode ser alterado arbitrariamente. Tais suposições são válidas para aplicações reais, tais como mineração de dados (como discutido em [26]), reconhecimento de padrões, *ray tracing*, simulações de Monte Carlo, e mapeamento de cromossomos. A classe de aplicações apresentada aqui não permite divisão arbitrária da computação entre diferentes processos, ou seja, não são consideradas tarefas com cargas de trabalho divisíveis [3]. De modo geral, uma aplicação BoT pode ser representada como um grafo bipartido em que um conjunto de nós representa os arquivos de entrada, o outro representa as tarefas, e as arestas representam dependência entre tarefas e arquivos [16] (Figura 1). Neste artigo, considera-se que os tamanhos dos arquivos de entrada e as dependências entre os arquivos e as tarefas são conhecidos.

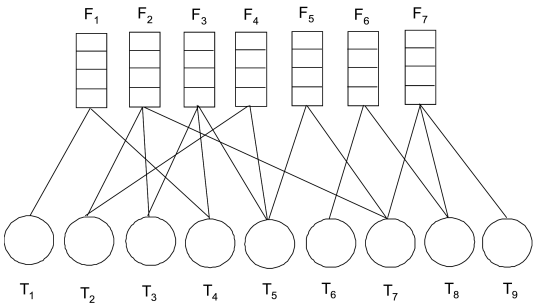


Figura 1. Aplicação BoT composta por 9 tarefas e 7 arquivos de entrada [11].

A plataforma mestre-escravo (ou *master-slave*, na língua inglesa) é uma abstração que representa a maioria das plataformas utilizadas para a execução de aplicações BoT [16, 2]. Em uma plataforma mestre-escravo, os nós escravos têm a função de executar

as tarefas da aplicação sob a supervisão centralizada de um nó mestre. Em plataformas mestre-escravo heterogêneas, nós (com múltiplos processadores) são organizados como uma árvore de dois níveis. O primeiro nível (topo) é composto por um nó mestre, que é responsável pelo escalonamento das tarefas entre os diversos nós escravos e coleta de resultados. No segundo nível da hierarquia, nós folha (escravos) apenas realizam computações. Uma vez que a plataforma é heterogênea, o poder computacional de cada nó escravo pode variar. Também considera-se que todos os arquivos de entrada são inicialmente armazenados no nó mestre, que serve como um repositório para todos os arquivos de entrada e de saída (contendo resultados). Cada arquivo de entrada requerido por uma tarefa tem que ser transferido do nó mestre para o nó escravo responsável pela execução da tarefa apenas se o arquivo ainda não existir nesse nó. Nós escravos possuem um disco local onde arquivos temporários podem ser armazenados. A transmissão de dados de novas tarefas do nó mestre para um determinado nó escravo pode ocorrer em paralelo com a execução de uma tarefa anteriormente escalonada para este nó.

3. Trabalhos Relacionados

Aplicações BoT compostas por tarefas independentes com compartilhamento de arquivos foram alvo de estudo em muitos trabalhos anteriores [2, 4, 9, 10]. Plataformas comumente utilizadas para executar aplicações BoT geralmente implementam o modelo mestre-escravo, que possui algumas limitações fundamentais: a comunicação do nó mestre e o acesso ao repositório de arquivos centralizado podem se transformar em gargalos para a execução das aplicações, e portanto, limitando a escalabilidade. Em [9], Casanova et al. propuseram heurísticas que consideram o compartilhamento de arquivos, de modo que os arquivos de entrada previamente transferidos para processadores não precisam ser retransferidos. Essa melhoria reduz o gargalo no nó mestre. Em [16], Giersch et al. provaram limites teóricos para a complexidade computacional associada a algumas instâncias do problema de escalonamento e propuseram várias novas heurísticas capazes de produzir escalonamentos que abordam o desempenho alcançado pelas heurísticas propostas por Casanova et al. [9], porém mantendo a complexidade computacional uma ordem de grandeza menor.

Em [20], Kumar e Rao propuseram a função de isoeficiência (*isoefficiency*, na língua inglesa) para caracterizar a escalabilidade de um algoritmo em uma determinada arquitetura. Essa função ajusta a eficiência até um certo valor desejado e calcula qual deve ser o aumento da quantidade de carga de trabalho para manter a eficiência inalterada na medida que o número de máquinas aumenta. Em [27], Sun et al. estudam a extensão da métrica de isovelocidade para sistemas computacionais heterogêneos. Os autores demonstram que a isovelocidade funciona bem em ambientes homogêneos e heterogêneos. No entanto, o estudo não se concentra nem em uma plataforma específica nem em uma classe de aplicações específica (por exemplo, aplicações BoT). Um trabalho semelhante foi realizado por Pastor e Bosque [22, 6, 5], propondo uma função de eficiência para sistemas computacionais heterogêneos. Seu trabalho amplia a noção de modelo de escalabilidade de isoeficiência homogêneo para sistemas de computação heterogêneos.

Anteriormente, foi estudada a escalabilidade de aplicações BoT com compartilhamento de arquivos em áreas como mineração de dados [26] e alinhamento de sequências (bioinformática) [15]. Em [11], foi desenvolvido um estudo mais teórico que demonstra que o limite inferior da função de isoeficiência de aplicações BoT executando em pla-

taformas mestre-escravo é $\Omega(P^2)$. Ou seja, para compensar o aumento dos custos de comunicação decorrentes do aumento do tamanho da plataforma e manter a eficiência em um patamar fixo, é preciso aumentar a carga de trabalho proporcionalmente a P^2, onde P é o número de nós. Nesse mesmo trabalho também foi proposto um algoritmo de escalonamento (chamado *Dynamic Clustering* - DC) que agrupa tarefas que compartilham arquivos e melhora a escalabilidade das aplicações. Buscando plataformas mais escaláveis, em [13, 12] foi realizado um estudo sobre plataformas hierárquicas, como extensões do mestre-escravo puro. Nesse trabalho, foi demonstrado que o limite de escalabilidade para a execução de aplicações BoT é $\Omega(P \log P)$ (ou seja, uma ordem de grandeza mais escalável do que o mestre-escravo puro) para vários modelos de comunicação incluindo *broadcast* e enlaces TCP). Em [25], foi avaliado o impacto da contenção causada pela transmissão de arquivos de saída na escalabilidade de aplicações BoT. Todos os estudos apresentados em [26, 11, 13, 25, 12, 15] trataram da escalabilidade de plataformas homogêneas. Contudo, grande parte das aplicações BoT é executada em plataformas heterogêneas [28]. Assim, o presente trabalho apresenta um estudo da escalabilidade de aplicações BoT considerando plataformas mestre-escravo compostas de nós heterogêneos. A computação heterogênea tem sido estudada por mais de duas décadas. Porém, poucos estudos avaliam os efeitos dessa heterogeneidade sobre a escalabilidade das aplicações.

Em [18], Yero e Henriques apresentam uma análise de escalabilidade de aplicações mestre-escravo em *clusters* heterogêneos. Inicialmente, os autores desconsideraram os efeitos da contenção de comunicação no modelo de execução, mas quando os autores explicitamente consideraram a contenção, eles concluíram que o sistema não é escalável quando a contenção é proporcional ao número de processadores. No presente estudo, é demonstrado que o par composto de plataformas mestre-escravo heterogêneas e aplicações BoT pode ser escalável, mesmo quando a contenção é proporcional ao número de processadores, desde que a quantidade de trabalho possa ser aumentada de acordo. Em [23, 24], Rosenberg e Chiang fizeram uma análise dos efeitos da heterogeneidade em plataformas distribuídas. As análises e resultados derivados deste trabalho são muito interessantes, com destaque para as seguintes conclusões: (i) se alguém pode substituir apenas um computador em um *cluster* por um mais rápido, é provável que (quase) sempre seja mais vantajoso substituir o mais rápido; (ii) se os computadores em dois *clusters* tiverem a mesma velocidade média, então o *cluster* com a maior variação na velocidade é (quase) sempre o mais rápido; (iii) heterogeneidade pode realmente dar poder a um *cluster*. Contudo, os autores não focam em uma arquitetura ou aplicação específicos.

O presente artigo apresenta um estudo sobre a escalabilidade de plataformas heterogêneas com foco em sistemas onde o poder de processamento dos nós de computação pode variar. Até onde se sabe, não há estudo que avalie a escalabilidade para a execução de aplicações BoT com compartilhamento de arquivos em plataformas mestre-escravo altamente heterogêneas como as analisadas neste trabalho.

4. Escalabilidade de Sistemas

A escalabilidade pode ser definida como a habilidade de um sistema para aumentar o *speedup* à medida que o número de processadores aumenta [17]. Outra definição diz que uma combinação entre algoritmo e máquina é escalável se a velocidade média alcançada pelo algoritmo na máquina em questão, permanecer constante com o aumento do número

de processadores, desde que o tamanho do problema possa ser aumentado com o tamanho do sistema [27]. Na equação 1, a eficiência é definida como a razão entre o *speedup* S e o número de processadores P, sendo que *speedup* pode ser definido pela razão entre o tempo necessário para a execução da carga de trabalho em um único processador (T_{seq}) e o tempo para a execução dessa mesma carga em P processadores (T_P).

$$E = \frac{S}{P} = \frac{\frac{T_{seq}}{T_P}}{P} \tag{1}$$

Em [20], Kumar e Chiang propuseram a função de isoeficiência para caracterizar a escalabilidade de um algoritmo em uma determinada arquitetura. A abordagem consiste em ajustar a eficiência até um certo valor desejado e calcular o quanto a carga de trabalho deve ser aumentada para manter a eficiência inalterada na medida que o número de máquinas aumenta. A função de isoeficiência $F(P)$ relaciona o tamanho da máquina P com a quantidade de trabalho W necessária para manter a eficiência. A quantidade de trabalho W é definida como a soma das quantidades de computação de todas as tarefas que compõem a aplicação. Isto é válido para plataformas homogêneas. Entretanto, a literatura apresenta uma proposta de eficiência heterogênea, que estende a análise de isoeficiência para máquinas heterogêneas [22, 6, 5]. A eficiência heterogênea é definida na equação 2.

$$E = \frac{W}{T_R \cdot C_T} \tag{2}$$

Na equação 2, W é a quantidade total de trabalho, T_R é o tempo de execução da aplicação e C_T é o poder computacional total da plataforma, definido como o somatório do poder computacional de cada máquina (C_i) que compõe a plataforma (equação 3). O poder computacional pode ser expresso em termos de alguma métrica de interesse, como por exemplo, a capacidade de realização de operações de ponto flutuante por segundo (FLOP/s).

$$C_T = \sum_{1}^{P} C_i \tag{3}$$

A escalabilidade pode ser obtida de forma horizontal (*scale-out*, na língua inglesa), adicionando-se nós ao sistema. Esse é o método mais comumente utilizado quando se utilizam recursos na nuvem [1]. A escalabilidade também pode ser implementada de forma vertical (*scale-up*, na língua inglesa), aumentando a quantidade de recursos (CPU, memória, velocidade de I/O, armazenamento) de um nó do sistema. A escalabilidade vertical também pode ser facilmente obtida por usuários de infraestrutura como serviço (IaaS), que podem escolher as configurações (e o preço) de suas instâncias.

5. Experimentos

Esta seção apresenta experimentos que comparam a escalabilidade de plataformas homogêneas e heterogêneas, usando a função de isoeficiência. Os experimentos foram feitos utilizando o Simgrid[3], que é um simulador amplamente utilizado na avaliação de sistemas distribuídos, tais como *clusters*, *grid*, sistemas *peer-to-peer* e nuvem [8]. Os experimentos foram executados em uma máquina virtual com 384 GB de memória e 40 VCPUs da nuvem da Universidade Federal de São Carlos (UFSCar) - Cloud@UFSCar[4].

[3] `http://simgrid.gforge.inria.fr/`

[4] `http://portalcloud.ufscar.br/`

5.1. Primeiro Experimento: Plataforma Homogênea *vs.* Heterogênea

O primeiro experimento tem por objetivo verificar se plataformas mestre-escravo heterogêneas podem apresentar ganhos na execução de aplicações BoT (em termos de escalabilidade) quando comparadas com plataformas homogêneas de mesma capacidade agregada. Para isso, foram simuladas plataformas mestre-escravo de diferentes tamanhos, e para cada tamanho, diferentes graus de heterogeneidade. Foram simuladas aplicações com tarefas homogêneas, de modo a analisar a escalabilidade em condições ideais. As aplicações homogêneas são compostas por tarefas com carga de trabalho de 3×10^{13} operações de ponto flutuante. Uma tarefa com esse tamanho leva cerca de 300 segundos para ser executada na máquina base de 100 GFLOP/s. O tamanho dos arquivos de entrada é 375 MB, sendo necessários cerca de 30 segundos para a transferência de um arquivo em um *link* de 100 Mbps. Ou seja, o tempo para executar uma tarefa na máquina de referência é cerca de dez vezes maior do que o tempo requerido para transmitir a tarefa (CCR - *Communication to Computation Ratio* = 0,1).

Os nós escravos têm a função de executar tarefas de uma aplicação BoT, e na prática podem ser implementados por nós de um *cluster*, máquinas em um *data center*, instâncias dedicadas (*bare-metal*) ou virtualizadas (máquinas virtuais ou contêineres) na nuvem. O nó mestre é responsável por despachar tarefas para execução nos nós escravos. Para maior realismo, alguns parâmetros do simulador foram calibrados com base no desempenho de uma plataforma real. Para modelar a capacidade da rede, foram feitas medidas com duas máquinas virtuais executando em diferentes máquinas físicas da Cloud@UFSCar, com o intuito de medir a largura de banda e a latência da rede que interliga os nós. Para medir a largura de banda foi utilizado o Iperf[5] e para medir a latência foi utilizado o Qperf[6]. As capacidades aferidas foram largura de banda média de 3,4 Gbps e uma latência média de 100 μs. A conexão entre o nó mestre e o nós escravos é representada por um *link* de 100 Mbps, que é uma velocidade típica oferecida por muitos provedores de *Internet*. Neste caso, considera-se que o nó mestre representa uma máquina externa de um usuário com acesso à plataforma distribuída, através da *Internet*.

A potência de 100 GFLOP/s para a máquina de referência foi adotada nos experimentos tendo em mente que processadores modernos oferecem desempenhos típicos no intervalo entre 15 a 75 GFLOP/s por núcleo em ponto flutuante de 64 bits [21]. Como o objetivo é medir a escalabilidade de plataformas heterogêneas, a potência da plataforma é expressa pela sua capacidade agregada (C_T). Neste experimento, a capacidade da plataforma variou de 100 GFLOP/s até 1 PFLOP/s, que seria equivalente a uma máquina com 2 núcleos até um *cluster* com cerca de 20000 núcleos. As plataformas foram geradas da seguinte forma:

- As plataformas homogêneas são geradas com 1, 10, 100, 1000 e 10000 nós com capacidade de 100 GFLOP/s. Nas Figuras 2 e 3, a curva de isoeficiência para plataformas homogêneas é rotulada como "0%" (0% de heterogeneidade).
- As plataformas heterogêneas são geradas com um percentual α de sua capacidade alocado em máquinas heterogêneas e $1-\alpha$ distribuídos igualmente entre um grupo de máquinas homogêneas. Os valores de α foram 25%, 50%, 75% e 99%.

[5] `https://iperf.fr/`

[6] `https://github.com/linux-rdma/qperf`

Com tarefas homogêneas e plataforma homogênea foi utilizado o escalonamento *round-robin*, que oferece desempenho ótimo [16]. No caso da plataforma heterogênea existe uma segunda etapa de escalonamento dinâmico que permite que nós ociosos roubem tarefas dos demais, balanceando a carga entre os nós. A cada experimento, a isoeficiência é estimada, aumentando-se a quantidade de trabalho necessária para atingir a eficiência desejada.

Os resultados são ilustrados na Figura 2. O gráfico está apresentado em escala *log-log* para evidenciar o comportamento assintótico de cada função de isoeficiência. O eixo horizontal contém o poder computacional agregado C_T da plataforma simulada, enquanto o eixo vertical representa o valor da função de isoeficiência. Vale destacar que a isoeficiência é dada pela quantidade de trabalho necessário para manter o nível de eficiência da aplicação BoT acima do patamar desejado, no caso 90%. Ou seja, quanto menor o valor (e a taxa de crescimento) da função, mais escalável é a plataforma.

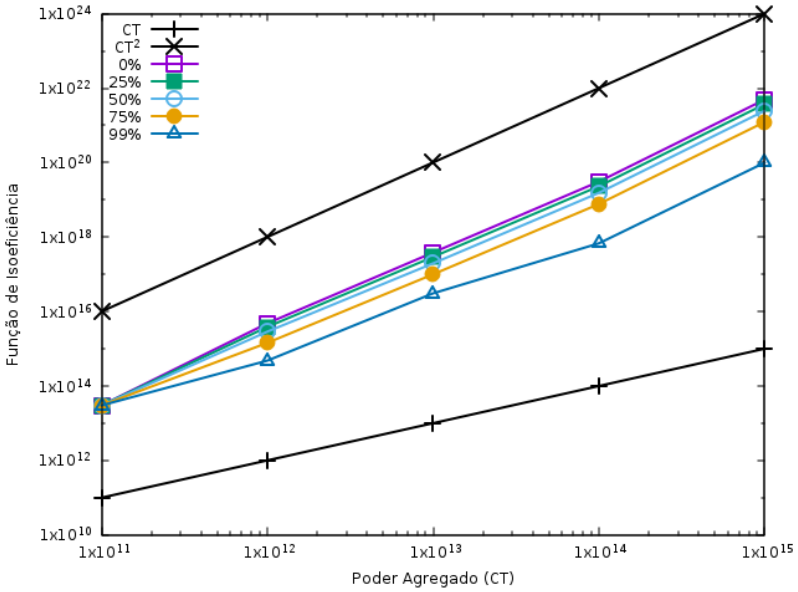


Figura 2. Função de isoeficiência para execução de tarefas homogêneas em plataformas mestre-escravo, mantendo a eficiência acima de 90%.

De maneira geral, verifica-se que a isoeficiência da plataforma homogênea (com rótulo de 0%) apresenta um comportamento assintótico similar (mesma taxa de crescimento) ao da curva ${C_T}^2$. Esse resultado é consistente com [11], onde foi provado que o limite inferior da função isoeficiência de plataformas mestre-escravo homogêneas é $\Omega(P^2)$. A novidade, contudo, é que as curvas de isoeficiência para plataformas heterogêneas apresentam crescimento inferior à medida que a heterogeneidade é aumentada. A plataforma que possui 99% de sua capacidade alocada em nós heterogêneos apresenta função de isoeficiência com crescimento inferior, bastante próximo ao da curva C_T. Nesse tipo de gráfico, mesmo pequenas diferenças de comportamento assintótico representam grandes diferenças em termos de desempenho da plataforma.

5.2. Segundo Experimento: Workloads Típicos de Sistemas Reais

O próximo conjunto de experimentos foi proposto com o objetivo de verificar se a heterogeneidade pode melhorar a escalabilidade de aplicações com perfis mais realistas, ou seja,

mais próximos das cargas de trabalho executadas cotidianamente em plataformas reais de produção ou de pesquisa. Para tal, foi utilizado um gerador de cargas sintéticas apresentado em [7], que reproduz as características (tempo entre submissão dos *jobs*, duração das tarefas, duração dos *jobs*, distribuição estatística) de cargas reais executadas em vários centros de supercomputação, segundo rastros de execução publicados no *Grid Workloads Archive* (GWA[7]). A Tabela 1 mostra a relação de coleções de rastros utilizados para a construção dos modelos de geração de carga.

Tabela 1. Bases de dados utilizadas no gerador de cargas. Adaptado de [7].

Sistema	Localização	Perfil
DAS-2	Holanda	Acadêmico
Grid'5000	França	Acadêmico
NorduGrid	Europa	Ambos
AuverGrid	França	Produção
SHARCNET	Canadá	Produção
EGEE/LCG	Europa	Produção

O procedimento para computar a função de isoeficiência é similar ao do primeiro experimento. São geradas as plataformas com diferentes tamanhos, variando de 100 GFLOP/s até 100 TFLOP/s. Fixa-se o nível de eficiência desejado (no caso, de 0,9) e em seguida geram-se cargas (*jobs* compostos de tarefas BoT) que são executadas até que seja atingido o patamar de eficiência desejado. As cargas são geradas de acordo com as distribuições de probabilidades verificadas em cada grupo de usuários para cada uma das seis plataformas listadas na Tabela 1. Os tamanhos dos arquivos são gerados de forma que o CCR médio seja de 0,1, tomando por base a máquina de referência.

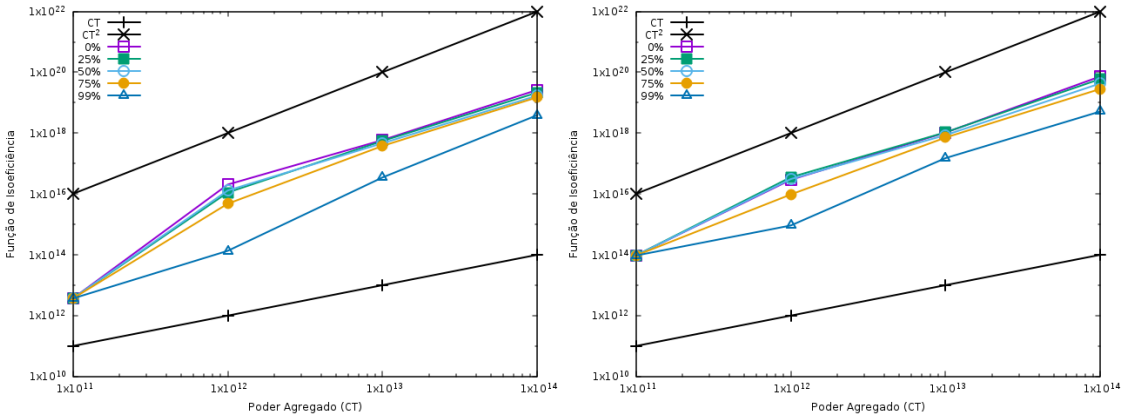
Os resultados são ilustrados na Figura 3. Como se pode observar em todos os casos avaliados, com ligeiras oscilações, as plataformas com mais heterogeneidade são mais escaláveis pois apresentam valores mais baixos para a função de isoeficiência. De fato, o experimento dá evidências de que o limite assintótico da função de isoeficiência de plataformas heterogêneas é inferior à curva ${C_T}^2$ (que é o limite assintótico de plataformas mestre-escravo homogêneas provado em [11]). Em outras palavras, substituir alguns nós de processamento por um único nó maior (de potência equivalente aos nós removidos) confere maior escalabilidade para a execução de aplicações BoT.

5.3. Terceiro Experimento: *Scale-out vs. Scale-up*

O objetivo do próximo experimento é verificar se empregar apenas a escalabilidade vertical para adicionar recursos a um sistema mestre-escravo é melhor do que utilizar a escalabilidade horizontal. Mais que isso, deseja-se identificar, em condições ideais, se existe diferença significativa (expressa pela curva de isoeficiência) no caso em que o número de nós é mantido e aumenta-se apenas as suas capacidades (*scale-up* puro) e o caso em que aumenta-se a capacidade adicionando nós homogêneos à plataforma.

Para isto, foi preciso criar um cenário que simula o comportamento de um sistema em condições ideiais para se obter máxima escalabilidade. O cenário criado foi uma

[7] http://gwa.ewi.tudelft.nl/



(a) DAS-2 (b) Grid'5000

(c) NorduGrid (d) AuverGrid

(e) SHARCNET (f) EGEE/LCG

Figura 3. Função de isoeficiência para execução de tarefas heterogêneas em plataformas mestre-escravo, mantendo a eficiência acima de 90%.

plataforma com um número fixo de 10 nós homogêneos e que vai gradativamente aumentando o poder de cada nó, de modo que a potência agregada da plataforma varia de 100 GFLOP/s a 10 PFLOP/s. Foram simuladas tarefas homogêneas de tamanho 3×10^{13} e 3×10^{12} FLOPs em um único *job* em que todas as tarefas dependem de um mesmo arquivo de entrada, pois estas seriam condições ideais para a melhor escalabilidade (*input file affinity* máximo), conforme provado em [11]. Foi utilizado o escalonamento *round-robin*,

que é ótimo [16]. Apesar de tais condições ideais serem improváveis em plataformas reais, este experimento tem por objetivo investigar uma questão de fundo teórico, mas que tem implicações práticas como será discutido adiante. Neste experimento o valor de eficiência foi fixado em 0,25 e em 0,5 (em vez de 0,9 utilizado nos experimentos anteriores). Isto possibilitou reduzir significativamente a quantidade de tarefas simuladas no experimento para se obter a curva de isoeficiência. Conforme demonstrado em [11, 13], fixar o nível de eficiência em patamares inferiores não altera o comportamento assintótico da curva de isoeficiência, mas apenas desloca a curva para baixo no gráfico *log-log*.

A Figura 4 mostra a curva de isoeficiência para tarefas de 3×10^{12} FLOPs. Nota-se que as duas curvas de eficiência de 0,5 e 0,25 apresentam tendência de crescimento muito abaixo de ${C_T}^2$ (limite inferior da função para plataformas mestre-escravo homogêneas provado em [11]) e até mesmo abaixo da curva $C_T \cdot \log C_T$ (limite inferior para plataformas hierárquicas provado em [13]). Mais que isso, as duas curvas apresentam uma tendência de crescimento bastante similar ao da curva C_T, sugerindo um desempenho de escalabilidade próximo ao linear das plataformas mestre-escravo quando se aumenta o poder da plataforma somente pelo *scale-up*.

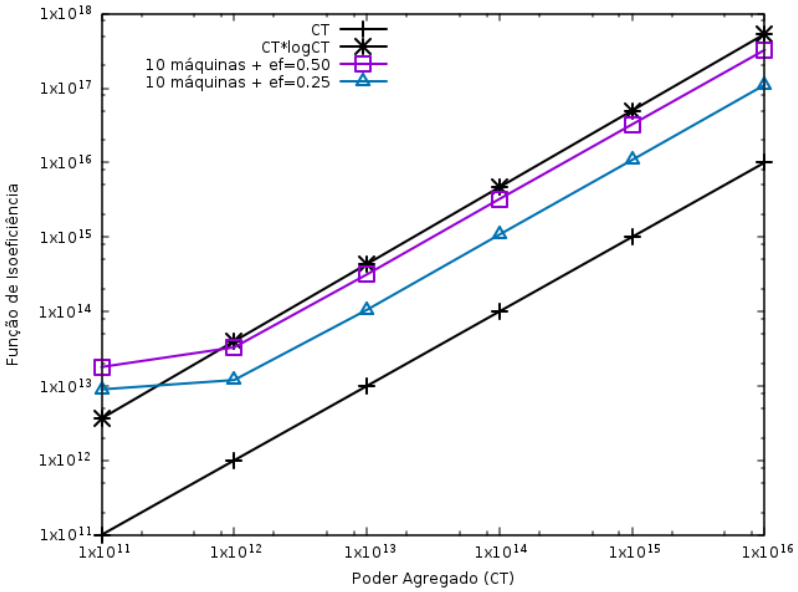


Figura 4. Função de isoeficiência para plataformas mestre-escravo homogênea, variando o poder dos nós.

5.4. Discussão

Um fator que afeta a eficiência, e consequentemente, limita a escalabilidade dos sistemas paralelos é o *overhead* de comunicação. O tempo de execução da aplicação é constituído pelo tempo efetivo para realizar as computações, somado ao tempo utilizado para comunicação e sincronização. No caso das aplicações BoT, o *overhead* decorre dos custos para coordenação e sincronização do sistema, bem como das transmissões de arquivos de entrada para os nós escravos e retorno dos resultados produzidos. As plataformas mestre-escravo homogêneas são bastante representativas do que ocorre na prática, quando usuários de nuvem fazem uso da elasticidade de recursos adicionando ou reduzindo nós à plataforma. Provavelmente pela simplicidade ou facilidade de implementação, essa tem

sido a forma típica e mais usual de se implementar a elasticidade de recursos, sendo que vários provedores oferecem serviços que automatizam o aumento ou redução do número de instâncias alocadas pelo cliente (em função da carga, do orçamento, etc). Contudo, a adição de nós aumenta também o *overhead* para transmissão dos dados e de coordenação da plataforma. Por outro lado, permitir que a plataforma possa ter recursos heterogêneos melhora sua escalabilidade, conforme demonstra o primeiro experimento. Ganhos significativos foram observados mesmo em cenários com cargas de trabalho típicas de plataformas reais, conforme mostrado no segundo experimento. Nas plataformas heterogêneas, é possível aumentar o poder computacional da plataforma substituindo nós existentes por nós de maior capacidade (escalabilidade vertical). Isto permite o crescimento da plataforma sem a adição de *overhead*, o que reduz o tempo de execução da aplicação e melhora a eficiência do sistema. Em plataformas heterogêneas, não é necessário que todos os nós sejam melhorados, mas apenas um ou alguns nós específicos. Em termos práticos, quando se executam aplicações BoT em uma plataforma como na nuvem, por exemplo, é quase sempre mais vantajoso substituir instâncias pequenas por uma única instância com capacidade equivalente. A escolha por instâncias de maior capacidade pode ser feita no momento inicial da execução da aplicação, ou posteriormente, à medida que o provedor disponibilize instâncias de maior capacidade computacional. De modo geral, sempre que houver disponibilidade, será mais vantajoso aumentar o poder da plataforma por meio do *scale-up*, e, quando este não for mais possível, aplicar então o *scale-out*. A presente análise não leva em conta o aumento dos custos das instâncias ou mesmo o orçamento para executar as aplicações. Isto será objeto de trabalhos futuros.

6. Conclusão e Trabalhos Futuros

O presente trabalho apresenta uma análise da escalabilidade de aplicações BoT executando em plataformas mestre-escravo heterogêneas. Tais plataformas são implementadas por grandes sistemas de computação distribuída, incluindo os *clusters*, grades computacionais, e pela nuvem. Foi demonstrado anteriormente que plataformas mestre-escravo homogêneas apresentam escalabilidade bastante limitada (com a função de isoeficiência crescendo proporcional a $\Omega({C_T}^2)$), enquanto que as plataformas hierárquicas homogêneas (que são uma extensão do mestre-escravo) são muito mais escaláveis, com isoeficiência crescendo proporcional a $\Omega(C_T \cdot \log C_T)$). Os resultados apresentados no presente trabalho demonstram, de forma experimental, que a heterogeneidade pode tornar a plataforma distribuída ainda mais escalável que as plataformas homogêneas citadas. Na prática, substituir um ou mais nós por um único de potência equivalente em geral traz melhoria de desempenho ao sistema. Os experimentos sugerem, ainda, que aumentar a capacidade do sistema por meio da escalabilidade vertical pode (sob certas condições), proporcionar escalabilidade próxima do linear. A prova formal deste resultado será objeto de trabalhos futuros. Por fim, este estudo considera apenas a escalabilidade do sistema e não leva em conta os custos envolvidos (como quando se substitui instâncias por outras de maior capacidade na nuvem). Isto será avaliado em trabalhos futuros.

Agradecimentos

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001. Fabricio A. B. Silva agradece à FAPERJ pelo apoio. Hermes Senger agradece ao CNPQ (Processo 305032/2015-1) e à FAPESP (Processos 2015/24461-2 e 2018/00452-2) pelo apoio.

Referências

[1] Yahya Al-Dhuraibi, Fawaz Paraiso, Nabil Djarallah, and Philippe Merle. Elasticity in Cloud Computing: State of the Art and Research Challenges. *IEEE Transactions on Services Computing (TSC)*, 11(2):430–447, March 2018.

[2] Olivier Beaumont, Larry Carter, Jeanne Ferrante, Arnaud Legrand, Loris Marchal, and Yves Robert. Centralized versus distributed schedulers for bag-of-tasks applications. *IEEE Transactions on Parallel and Distributed Systems*, 19(5):698–709, 2008.

[3] Olivier Beaumont, Arnaud Legrand, and Yves Robert. Scheduling divisible workloads on heterogeneous platforms. *Parallel Computing*, 29(9):1121–1152, 2003.

[4] Francine Berman, Richard Wolski, Henri Casanova, Walfredo Cirne, Holly Dail, Marcio Faerman, Silvia Figueira, Jim Hayes, Graziano Obertelli, Jennifer Schopf, et al. Adaptive computing on the grid using apples. *IEEE Transactions on Parallel and Distributed Systems*, 14(4):369–382, 2003.

[5] Jose L. Bosque, Oscar D. Robles, Pablo Toharia, and Luis Pastor. Evaluating scalability in heterogeneous systems. *The Journal of Supercomputing*, 58(3):367–375, Dec 2011.

[6] Jose Luis Bosque and Luis Pastor Perez. Theoretical scalability analysis for heterogeneous clusters. In *IEEE International Symposium on Cluster Computing and the Grid, 2004. (CCGrid)*, pages 285–292. IEEE, 2004.

[7] M. Carvalho and F. Brasileiro. A user-based model of grid computing workloads. In *2012 ACM/IEEE 13th Intl. Conf. on Grid Computing*, pages 40–48, Sept 2012.

[8] Henri Casanova, Arnaud Giersch, Arnaud Legrand, Martin Quinson, and Frédéric Suter. Versatile, scalable, and accurate simulation of distributed applications and platforms. *Journal of Parallel and Distributed Computing*, 74(10):2899–2917, June 2014.

[9] Henri Casanova, Arnaud Legrand, Dmitrii Zagorodnov, and Francine Berman. Heuristics for scheduling parameter sweep applications in grid environments. In *9th IEEE Heterogeneous Computing Workshop.(HCW 2000)*, pages 349–363. IEEE, 2000.

[10] Walfredo Cirne, Daniel Paranhos, Lauro Costa, Elizeu Santos-Neto, Francisco Brasileiro, Jacques Sauve, Fabricio AB Silva, Carla O Barros, and Cirano Silveira. Running bag-of-tasks applications on computational grids: The mygrid approach. In *International Conference on Parallel Processing (ICPP)*, pages 407–416. IEEE, 2003.

[11] Fabricio A. B. da Silva and Hermes Senger. Improving scalability of bag-of-tasks applications running on master–slave platforms. *Parallel Computing*, 35(2):57–71, 2009.

[12] Fabricio A. B. da Silva and Hermes Senger. Scalability analysis of large distributed computing systems. In *Grid Computing: techniques and future prospects*. Nova Science Publishers, 2015.

[13] Fabricio A.B. da Silva and Hermes Senger. Scalability limits of bag-of-tasks applications running on hierarchical platforms. *J. Parallel and Distributed Computing*, 71(6):788–801, 2011.

[14] Rafael Ferreira da Silva and Tristan Glatard. A science-gateway workload archive to study pilot jobs, user activity, bag of tasks, task sub-steps, and workflow executions. In *European Conference on Parallel Processing*, pages 79–88. Springer, 2012.

[15] Marcelo Rodrigo de Castro, Catherine dos Santos Tostes, Alberto MR Dávila, Hermes Senger, and Fabricio AB da Silva. Sparkblast: scalable blast processing using in-memory operations. *BMC bioinformatics*, 18(1):318, 2017.

[16] Arnaud Giersch, Yves Robert, and Frédéric Vivien. Scheduling tasks sharing files on heterogeneous master–slave platforms. *J. Systems Architecture*, 52(2):88–104, 2006.

[17] Ananth Y Grama, Anshul Gupta, and Vipin Kumar. Isoefficiency: Measuring the scalability of parallel algorithms and architectures. *IEEE concurrency*, 1(3):12–21, 1993.

[18] Eduardo Javier Huerta Yero and Marco Aurélio Amaral Henriques. Speedup and scalability analysis of master–slave applications on large heterogeneous clusters. *Journal of Parallel and Distributed Computing*, 67(11):1155–1167, 2007.

[19] Alexandru Iosup, Ozan Sonmez, Shanny Anoep, and Dick Epema. The performance of bags-of-tasks in large-scale distributed systems. In *Proc.17th International Symposium on High Performance Distributed Computing*, pages 97–108. ACM, 2008.

[20] Vipin Kumar and V Nageshwara Rao. Parallel depth first search. part ii. analysis. *International Journal of Parallel Programming*, 16(6):501–519, 1987.

[21] Simon McIntosh-Smith, James Price, Tom Deakin, and Andrei Poenaru. Comparative benchmarking of the first generation of hpc-optimised arm processors on isambard. *Concurrency and Computation Practice and Experience - Special Issue on the Cray User Group*, pages –, 2018 (to appear).

[22] Luis Pastor and José L Bosque. An efficiency and scalability model for heterogeneous clusters. In *Intl. Conf. Cluster Computing (CLUSTER)*, pages 427–427. IEEE, 2001.

[23] Arnold L Rosenberg and Ron C Chiang. Toward understanding heterogeneity in computing. In *Intl. Symp. Parallel & Dist. Processing (IPDPS)*, pages 1–10. IEEE, 2010.

[24] Arnold L Rosenberg and Ron C Chiang. Heterogeneity in computing: Insights from a worksharing scheduling problem. *International Journal of Foundations of Computer Science*, 22(06):1471–1493, 2011.

[25] Hermes Senger and Fabricio A. B. da Silva. Bounds on the scalability of bag-of-tasks applications running on master-slave platforms. *Paral. Proc. Letters*, 22(02), 2012.

[26] Hermes Senger, Eduardo R Hruschka, Fabrício AB Silva, Liria M Sato, Calebe P Bianchini, and Bruno F Jerosch. Exploiting idle cycles to execute data mining applications on clusters of pcs. *Journal of Systems and Software*, 80(5):778–790, 2007.

[27] Xian-He Sun and Diane T Rover. Scalability of parallel algorithm-machine combinations. *IEEE Transactions on Parallel and Distributed Systems*, 5(6):599–613, 1994.

[28] Long Thai, Blesson Varghese, and Adam Barker. A survey and taxonomy of resource optimisation for executing bag-of-task applications on public clouds. *Future Generation Computer Systems*, 82:1 – 11, 2018.