Explorando a Plataforma de Computação em Nuvem Heroku para Execução de Programas Paralelos com OpenMP

Ana Luísa V. Solórzano[1], **Andrea S. Charão**[1]

[1] Laboratório de Sistemas de Computação (LSC)
Universidade Federal de Santa Maria (UFSM)

Resumo. *Neste trabalho, explora-se os recursos da plataforma Heroku, uma popular solução na modalidade de "plataforma como serviço" (*Platform as a Service – *PaaS), na execução de programas paralelos com OpenMP. A partir de experimentos com um programa simples, executado na nuvem e em um servidor real, nota-se que a plataforma apresenta grande variabilidade no desempenho de um programa com OpenMP, mesmo sem suporte oficial a esta modalidade.*

1. Introdução

A computação em nuvem é um paradigma que tem motivado muitos trabalhos de pesquisa, assim como muitos negócios em Tecnologia da Informação. Este paradigma se caracteriza pelo acesso a recursos computacionais sob demanda, via rede, com esforço mínimo de gerenciamento e sem custos de aquisição de hardware [Vaquero et al. 2008, Mell e Grance 2011]. Muitas empresas atualmente oferecem serviços de computação em nuvem, em diversas modalidades e com diferentes modelos de precificação, geralmente voltados à hospedagem de aplicações de sistemas de informação.

A área de computação de alto desempenho também se beneficia desse paradigma, embora não seja o alvo mais comum dos provedores de serviços em nuvem. De fato, demandas de alto desempenho geralmente sugerem recursos dedicados, o que vai num sentido contrário ao modelo em nuvem, em que recursos são virtualizados e compartilhados. Mesmo assim, há trabalhos que exploram as vantagens e desvantagens deste paradigma para executar aplicações paralelas e/ou distribuídas, desenvolvidas com técnicas de computação de alto desempenho [Alves e de Assumpção Drummond 2014, Silva e de Oliveira 2016].

Pesquisas exploratórias nesta área costumam focar em serviços na modalidade de "infraestrutura como serviço" (*Infrastructure as a Service* – IaaS), oferecida, por exemplo, pelas plataformas Amazon EC2 ou Microsoft Azure. No entanto, pode-se estimar que outras modalidades e fornecedores de serviços também se prestem à execução de aplicações paralelas e/ou distribuídas. Neste trabalho, explora-se os recursos da plataforma Heroku, uma popular solução na modalidade de "plataforma como serviço" (*Platform as a Service* – PaaS), em um cenário típico de computação de alto desempenho: a execução de programas paralelos com OpenMP. Embora não oficialmente suportado pela plataforma, toma-se por hipótese que este cenário seja viável e apresente vantagens, com possíveis limitações.

2. Plataforma Heroku

O Heroku[1] é uma plataforma criada em 2007 que disponibiliza um ambiente de computação em nuvem com suporte a várias linguagens de programação, permitindo ao usuário criar e submeter suas aplicações.

[1] Disponível em: https://www.heroku.com

Os aplicativos criados são executados em contêineres Unix isolados e virtualizados, chamados *dynos*, que fornecem o ambiente de execução necessário, lidando com requisições web e processando trabalhos em segundo plano. Dentre os tipos de ambiente disponíveis, existem os *one-off-dynos*, que são *dynos* temporários criados para executar comandos e tarefas esporádicas. Esse tipo de *dyno* permite, por exemplo, executar uma sessão interativa com um interpretador de comandos na nuvem Heroku.

Ao ser criado, cada aplicativo é associado a um repositório remoto, sendo via Git a principal forma de lançá-lo à plataforma. Para isso, passa-se o código fonte, uma descrição das dependências utilizadas e um arquivo (`Procfile`) contendo o(s) comando(s) que o Heroku deve utilizar para executar a aplicação.

A plataforma oferece planos pagos e gratuitos, dependendo do serviço requerido. O plano gratuito oferece acesso a um *dyno* e tem a conexão interrompida após trinta minutos de inatividade; já os planos pagos oferecem acesso a mais *dynos* e outras ferramentas que auxiliam no desenvolvimento, e o plano mais caro oferece inclusive acesso a uma máquina dedicada. Mesmo no plano gratuito, dispõe-se de um *dyno* com várias CPUs/*threads* que, mesmo não dedicadas, sugerem que o ambiente possa ser usado para executar aplicações concorrentes e paralelas.

3. OpenMP na Plataforma Heroku

O compilador do Heroku é responsável por detectar o tipo de aplicação a ser submetida, buscar as dependências necessárias para a execução do programa, compilar o código utilizando um compilador nativo, caso necessário, e o lançar para um *dyno*. Atualmente, a plataforma disponibiliza um compilador GCC 4.8.4 nativo, com suporte a OpenMP.

Para gerar programas executáveis em um *dyno*, são utilizados *buildpacks*. Os *buildpacks* são compostos por conjuntos de *scripts* que identificam a linguagem de programação utilizada para poder compilar o código, recuperando dependências. Deve-se atentar para a compilação de programas com muitas dependências, pois pode levar à falha na execução por ausência de dependências não reconhecidas. Uma solução para esse caso seria compilar o programa com bibliotecas estáticas em uma máquina real e depois passar o executável para a nuvem.

Por padrão, existe uma lista de *buildpacks* oficiais disponíveis, que são verificados automaticamente ao importar um código ao Heroku. Entretanto, *buildpacks* não oficiais também podem ser associados através do link para o seu repositório Git. O *buildpack* usado neste trabalho destina-se a programas em C[2].

4. Experimentos

Para analisar a eficiência do Heroku como um ambiente de processamento de alto desempenho, em comparação a uma máquina real, utilizou-se um programa em C[3] que conta os números primos em um intervalo de 1 a N.

O programa realiza o cálculo verificando quais números menores do que N são seus divisores, tendo um total de trabalho aproximadamente proporcional a $1/2 \times N^2$. O processamento paralelo ocorre utilizando diretivas OpenMP em um laço *for* que divide os

[2] *Buildpack* utilizado nos experimentos: https://github.com/heroku/heroku-buildpack-c.git

[3] Disponível em: https://people.sc.fsu.edu/ jburkardt/c_src/prime_openmp/prime_openmp.html

cálculos entre as *threads*. Um dos motivos para ser usado nos experimentos foi o fato dele ter menos dependências do que *benchmarks* consagrados e mesmo assim levar um tempo considerável de execução para o caso observado, de N=500.000.

A máquina real utilizada para as execuções foi um servidor localizado no Laboratório de Sistemas de Computação (LSC) da UFSM, que possui um processador Intel® Xeon® E5620 de quatro cores físicos e oito virtuais, com 32KB de cache L1, 256KB de cache L2 e 12MB de cache L3, e 12 GB de memória. O sistema operacional da máquina é Debian 8 e versão do Linux 3.2.73-2+deb7u2.

O *dyno* utilizado no Heroku corresponde à modalidade gratuita, que apresenta-se como uma máquina virtual Xen executando sobre um processador Intel(R) Xeon(R) CPU E5-2670 de oito cores, com cache L1 de 32KB, L2 de 256KB e L3 de 20MB e 60GB de memória RAM. O sistema operacional é Ubuntu e versão do Linux 3.13.0-105-generic. Foram feitas 30 execuções com o programa em OpenMP, variando-se o número de *threads* entre 1, 2, 4 e 8, resultando em 120 experimentos sobre os quais foram feitos cálculos estatísticos e de desempenho.

5. Resultados e Discussão

As tabelas 1 e 2 apresentam estatísticas sobre os experimentos com o Heroku e com o servidor do LSC. A partir dos dados obtidos, observou-se que as execuções na nuvem geraram resultados com ampla variabilidade e ampla margem de erro para um intervalo de confiança de 99%. Esses resultados estão vinculados ao fato de não poder ser garantido o acesso exclusivo à nuvem, já que os recursos são compartilhados entre diversas máquinas virtuais. Resultados similares foram observados em uma análise de desempenho da Amazon EC2, relatados em [Schad et al. 2010]. Apesar disso, algumas execuções apresentaram tempos baixos, condizentes com a especificação da máquina virtual.

Thread	Média	Variância	Desvio padrão	Coeficiente de variação	Tempo mínimo	Tempo máximo	Margem de erro (99%)
1	53,800	106,960	10,342	0,192	19,096	80,329	3,806
2	43,747	161,423	12,705	0,290	20,534	82,698	4,675
4	41,269	239,306	15,470	0,375	21,128	86,640	5,692
8	48,645	422,833	20,563	0,423	17,499	84,516	7,567

Tabela 1. Tempos de execução (segundos) na plataforma Heroku

Thread	Média	Variância	Desvio padrão	Coeficiente de variação	Tempo mínimo	Tempo máximo	Margem de erro (99%)
1	41,216	0,002	0,039	0,001	41,196	41,403	0,0186
2	30,414	0,003	0,051	0,002	30,377	30,597	0,024
4	18,041	0,006	0,075	0,004	17,957	18,190	0,035
8	12,800	0,223	0,473	0,037	12,017	14,134	0,222

Tabela 2. Tempos de execução (segundos) no servidor do LSC

A tabela 3 mostra a aceleração para os experimentos em cada ambiente na coluna *speedup*, em função do número de *threads*. A partir dela, observou-se que o desempenho

da aplicação não explora de forma eficiente o paralelismo, pois mesmo no servidor do LSC não obteve alta eficiência.

Thread	Speedup Heroku	Eficiência Heroku	Speedup lsc4	Eficiência lsc4
2	1,230	0,615	1,355	0,678
4	1,304	0,326	2,2846	0,571
8	1,106	0,138	3,220	0,402

Tabela 3. Análise de desempenho

6. Considerações Finais

Neste trabalho, observou-se que a execução de programas com OpenMP no serviço gratuito da nuvem Heroku não se presta bem a experimentos de avaliação de desempenho, pois a variabilidade dos tempos de execução é muito significativa, justificada pelo compartilhamento da máquina física provida pela plataforma. Por outro lado, embora não ofereça suporte oficial à execução de programas paralelos com OpenMP, é possível desempenhar esta tarefa com bastante agilidade, sem necessidade de se administrar um servidor. Isso pode se prestar a exercícios e testes que não exijam medições de tempo. Os resultados obtidos também motivam outros testes em trabalhos futuros, com aplicações que se comportem melhor num ambiente real.

Referências

Alves, M. M. e de Assumpção Drummond, L. M. (2014). Análise de desempenho de um simulador de reservatórios de petróleo em um ambiente de computação em nuvem. In *Anais do 15º Simpósio em Sistemas Computacionais de Alto Desempenho (WSCAD-SSC'14)*.

Mell, P. e Grance, T. (2011). The nist definition of cloud computing. Technical Report 800-145, National Institute of Standards and Technology (NIST).

Schad, J., Dittrich, J., e Quiáne-Ruiz, J.-A. (2010). Runtime measurements in the cloud: Observing, analyzing, and reducing variance. *Proceeding of the VLDB Endowment*, 3(1–2):460–471.

Silva, A. L. e de Oliveira, T. B. (2016). Avaliação de escalabilidade de aplicações de alto desempenho em núvem pública e privada. In *Anais do 17º Simpósio em Sistemas Computacionais de Alto Desempenho (WSCAD-SSC'16)*.

Vaquero, L. M., Rodero-Merino, L., Caceres, J., e Lindner, M. (2008). A break in the clouds: Towards a cloud definition. *SIGCOMM Comput. Commun. Rev.*, 39(1):50–55.