Uma Ferramenta de Software para a Predição de Desempenho de Workflows Científicos

Aluno: Lucas Magno
Bolsista PIBIC da CNPq
Instituto de Física (IF)

Orientadora: Kelly Rosa Braghetto
Departamento de Ciência da Computação (DCC)
Instituto de Matemática e Estatística (IME)

Universidade de São Paulo
lucas.magno@usp.br

Resumo

Este documento descreve as atividades realizadas durante o período de julho de 2013 a junho de 2014 no âmbito do projeto de iniciação científica do aluno Lucas Magno, número USP 7994983, orientado pela Profa. Dra. Kelly Rosa Braghetto e financiado por uma bolsa PIBIC/CNPq.

O objetivo principal do projeto foi desenvolver uma ferramenta de software para a conversão automática de modelos de *workflows* em modelos estocásticos na álgebra de processos *PEPA - Performance Evaluation Process Algebra* [4] . A partir desses modelos estocásticos, é possível extrair predições de desempenho de *workflows*.

Abstract

Introdução

Inicialmente desenvolvidos para automatizar processos industriais e empresariais, os *workflows* se popularizaram e passaram a ser usados na modelagem e automatização de experimentos científicos em diversas áreas da ciência. Um *workflow científico* é a descrição completa ou parcial de um experimento científico em termo de suas atividades, controles de fluxo e dependência de dados [8].

Há várias maneiras de se representar um *workflow científico*, entre elas *grafos direcionados*, *UML* (Unified Modeling Language), *redes de Petri* e *álgebras de processo* [9]:. Estes mecanismos de representação são usados para criar modelos que especificam a ordem de execução das atividades dos *workflows*. Além disso, as redes de Petri e as álgebras de processo são linguagens formais, permitindo que se verifiquem propriedades qualitativas e quantitativas dos modelos de *workflow*. Neste trabalho, no entando, somente foram utilizados grafos direcionados e álgebras de processo.

Para simplificar sua implementação, considera-se que *workflows* sejam compostos por *atividades*, que representam atividades reais de um experimento, e estruturas para descrever o fluxo de controle, como *sequência*, *paralelismo*, *escolha* e *sincronização*, definidas por meio dos operadores *AND* (paralelismo/sincronização), *XOR* (escolha exclusiva/junção) e *OR* (escolha múltipla/junção).

Por ser comum em experimentos científicos a manipulação de de enormes quantidades de dados e a presença de processos muito demorados, é necessária a análise do desempenho dos *workflows* associados, que pode ser feita através de *medição*, *simulação* ou *modelagem analítica* [7]. Foi escolhida, então, a modelagem analítica, um método preditivo e rápido, implementada por meio de uma álgebra de processos estocástica, a *PEPA*, pois seu uso ainda não foi profundamente explorado para a análise de desempenho preditiva de *workflows* científicos.

Objetivos

Uma desvantagem da modelagem analítica usando *PEPA* é a necessidade da descrição do *workflow* em uma linguagem de modelagem estocástica e utilização de programas específicos para a análise, exigindo do usuário um certo nível de conhecimento sobre álgebras de processo. No entanto, *workflows* científicos são utilizados em diversas áreas da ciência que não necessitam de um grande aprofundamento em computação, o que pode inviabilizar a aplicação deste método.

O objetivo do trabalho foi facilitar a extração de predições de desempenho para *workflows* científicos. Para isso, foi desenvolvida uma ferramenta de software que gera modelos estocásticos em *PEPA* e sua solução numérica a partir de modelos de *workflows*.

Os modelos de *workflows* usados como entrada para a ferramenta são descritos textualmente na forma de um grafo dirigido - uma representação simples e que pode ser usada com facilidade por usuários não especialistas. Além do modelo em *PEPA* e sua solução, a ferramenta também gera uma representação gráfica do modelo de *workflow*, que permite que o usuário possa verificá-lo mais facilmente.

Materiais e Métodos

Para que possua um modelo correspondente em *PEPA*, um modelo de *workflow* precisa ser bem estruturado e não possuir ambiguidades semânticas. Por essa razão, neste trabalho consideramos apenas modelos de *workflow* que apresentam somente um ponto de entrada e um ponto de saída, têm sua estrutura em forma de "blocos" e não apresentam ciclos, ou laços, o que permite uma implementação mais simples.

Para automatizar o processo de predição de desempenho, foi implementado um programa na linguagem *Python* (versão 2.7), por flexibilidade, facilidade de aprendizado e grande número de bibliotecas auxiliares. Seu código fonte, detalhes de sua execução, dependências e exemplos de *workflow* de entrada e suas representações em *PEPA* geradas podem ser conferidos na página do projeto [1].

O programa pode ser resumido nas seguintes etapas:

1. Lê como entrada uma descrição textual de um *workflow* em uma gramática simples baseada na linguagem *DOT* [2] através dos analisadores léxico e sintático gerados a partir da biblioteca *PLY*, *Python Lex-Yacc* [5].

2. Gera uma estrutura de dados baseada em grafo na memória representando o *workflow* através de classes explicitamente definidas que permitem a manipulação de nós, arestas e *workflows*.

3. Gera uma uma descrição do *workflow* de entrada em linguagem *DOT* e sua visualização através da biblioteca *Graphviz* [3].

4. Gera um modelo analítico do *workflow* em *PEPA*

5. Gera a solução numérica do modelo analítico e extrai seus índices de desempenho através da biblioteca *pyPEPA* [6], uma implementação recente da *PEPA* em *Python*.

Resultados

Ao processar um *workflow*, então, o programa cria arquivos de saída contendo a descrição do *workflow* em *DOT*, sua visualização, sua descrição em *PEPA* e os índices de desempenho dela extraídos. Estes arquivos, entretanto, são geralmente muito extensos, então serão mostrados aqui somente um exemplo de *workflow* de entrada e sua visualização. Demais exemplos estão disponíveis na página do projeto.

```
1  digraph {
2      a              -> b;
3      b              -> and1;
4      and1 [and]     -> e, xor1;
5      xor1 [xor]     -> [0.15] c, [0.85] d;
6      e [0.5]        -> and2;
7      c              -> xor2;
8      d              -> xor2;
9      xor2           -> and2;
10     and2           -> f;
11 }
```

Código 1: Exemplo de descrição do *workflow* de entrada

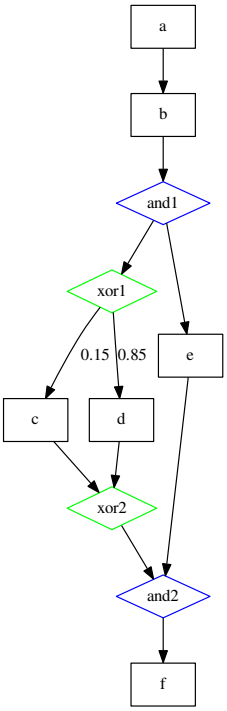

Figura 1: Exemplo de visualização do *workflow* criada a partir do código em DOT

Conclusões

Referências

[1] *Código fonte da ferramenta de software desenvolvida, exemplos de workflow de entrada e seus respectivos modelos em PEPA*. www.ime.usp.br/~kellyrb/ic/#lucas.

[2] *The DOT Language | Graphviz - Graph Visualization Software*. http://www.graphviz.org/content/dot-language.

[3] *Graphviz | Graphviz - Graph Visualization Software*. http://www.graphviz.org/.

[4] *PEPA - Performance Evaluation Process Algebra*. http://www.dcs.ed.ac.uk/pepa/.

[5] *PLY (Python Lex-Yacc)*. http://www.dabeaz.com/ply/.

[6] *pypepa - Python toolset for PEPA*. https://github.com/tdi/pyPEPA.

[7] Braghetto, K. R.: *Técnicas de Modelagem para a Análise de Desempenho de Processos de Negócio*. Tese de Doutoramento, Instituto de Matemática e Estatística da Universidade de São Paulo, 2011.

[8] Gadelha, L. M. R.: *Gerência de Proveniência em Workflows Científicos Paralelos e Distribuídos*. Tese de Doutoramento, Universidade Federal do Rio de Janeiro, 2012.

[9] Ogasawara, E. S.: *Uma Abordagem Algébrica para Workflows Científicos com Dados em Larga Escala*. Tese de Doutoramento, Universidade Federal do Rio de Janeiro, 2011.