Modelagem e simulação de relações entre tarefas computacionais em ambiente de nuvem*

Luís Vinícius Baldissera[1], **Aleardo Manacero Jr.**[1], **Renata Spolon Lobato**[1]
Roberta Spolon[2]

[1]Departamento de Ciências de Computação e Estatística
Universidade Estadual Paulista "Júlio de Mesquita Filho" (UNESP)

[2]Departamento de Computação
Universidade Estadual Paulista "Júlio de Mesquita Filho" (UNESP)

Abstract. *Cloud computing enabled computational loads composed by dependent tasks, in a workflow structure. These dependence relationships demand specific characteristics in the resource management algorithms, which efficiency can be evaluated through simulation. However, the simulators normally used to test such methods, either do not support the workflow modeling, or require knowledge of some programming language from their user. Thus, it is proposed in this work a simple and robust graphic approach to model interdependencies between tasks.*

Resumo. *A computação em nuvem permite cargas computacionais contendo tarefas dependentes entre si, numa estrutura de workflow. Essas relações de dependência demandam características específicas em algoritmos para gerenciamento de recursos, cuja eficiência pode ser avaliada por simulação. No entanto os simuladores que são normalmente utilizados para testar tais métodos, ou não dão suporte à modelagem de workflow ou exigem do usuário conhecimento de alguma linguagem de programação. Assim, este trabalho propõe uma abordagem gráfica, robusta e simples, para modelagem de interdependências entre as tarefas.*

1. Introdução

Sistemas de computação em nuvem oferecem serviços para o desenvolvimento de aplicações mais escaláveis, as quais podem ser vistas como componentes de uma estrutura de tarefas interdependentes, ou *workflow*. Dessa forma, a constante busca por qualidade de serviço demandam abordagens mais eficientes ou adaptadas, capazes de serem avaliadas por simulação, evitando medições no sistema real, tornando fundamental a modelagem de *workflows* para esses sistemas.

No entanto, os principais simuladores de nuvem não oferecem modelagem de *workflow* de forma intuitiva, ou exigem do usuário o conhecimento de alguma linguagem de programação para implementá-la, como é o caso do CloudSim [Calheiros et al. 2011]. Este trabalho apresenta uma abordagem intuitiva e robusta, baseada em ícones, para representar as diferentes relações entre as tarefas de um *workflow*.

A seguir é apresentada uma revisão sobre o estado da arte na área, seguida da metodologia proposta para definir e implementar o modelo e conclusões sobre o trabalho.

*processo nº 2019/06281-8, Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP)

2. Trabalhos Relacionados

A literatura, no que diz respeito à modelagem de interdependências entre tarefas, traz predominantemente o uso de DAG (Grafo Acíclico Direcionado), como utilizado por Satish et al [Satish et al. 2008], que tem como vantagem a intuitividade de sua visualização.

Tick [Tick 2006] apresenta diferentes métodos para representação do *workflow*, que permitem também a modelagem de ciclos, como Rede de Petri, UML e Diagrama de Roteamento. Já a notação LOTOS [Bolognesi and Brinksma 1987], que é uma formalização de interconexão de sistemas, se mostra robusta quanto a representação dos tipos de dependências e trocas de mensagem, mas não é tão intuitiva de ser visualizada.

Este trabalho propõe um novo modelo baseado na estrutura de DAG, que abstrai algumas das relações estabelecidas pela notação formal LOTOS, balanceando a simplicidade de visualização e a robustez de representação.

3. Metodologia

Para a representação gráfica de *workflows* define-se um DAG, no qual os vértices representam tipos de carga de trabalho atômica e as arestas as dependências entre os elementos da carga, de acordo com algumas das relações da notação LOTOS, como descrito na Tabela 1.

3.1. Componentes para modelos de carga

Tabela 1. Ícones do DAG de acordo com sua descrição

Vértices	
	**Tarefa** – representa a configuração de uma tarefa com sua carga computacional, incluindo dependências
	**Espera** – representa uma espera de um determinado tempo
	**Sincronização** – representa um ponto de sincronização, e é concluído quando todas suas dependências forem concluídas
	**Ativação** – representa um ponto de ativação, sendo concluído quando qualquer de suas dependências for concluída
	**Desvio** – representa um desvio de fluxo na execução das tarefas, no qual cada fluxo tem uma probabilidade associada
Arestas	
	**Precedência** – o predecessor deve ser concluído antes do sucessor iniciar
	**Prefixação** – o prefixo deve ser iniciado antes do sufixo ser iniciado.
	**Tratamento de Falha** – o fluxo de execução pode ser desviado para o tratamento de uma falha.

Além dos vértices e arestas apresentados na Tabela 1, existem também nós de conveniência, denominados expansões e descritos na Tabela 2, que permitem a modelagem de cargas mais complexas, como ciclos e recursões, utilizando poucos itens, como o *workflow* LIGO [Ramakrishnan et al. 2007], que pode ser observada na Figura 1 .

Tabela 2. Expansões do DAG

Ícone	Descrição
	**Iterativa** – representa a execução do DAG interno iterativamente dado um número de repetições
	**Paralela** – representa a execução do DAG interno em n vias paralelas, concluídas como ponto de sincronização ou ativação
	**Recursiva** – representa a execução recursiva do DAG interno por n vezes

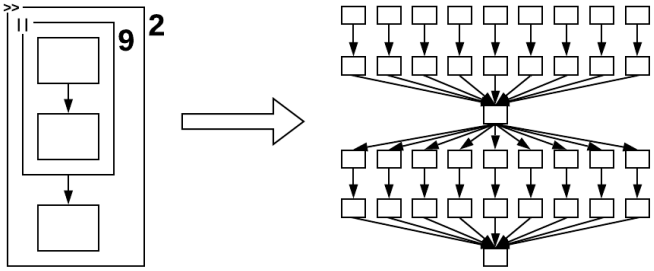

Figura 1. Exemplo de modelagem. Workflow adaptado de Ramakrishnan (2007)

3.2. Arquitetura para simulação das interações

Este modelo será implementado no iSPD, um simulador de sistemas paralelos e distribuídos, capaz de simular ambiente de nuvem [Manacero et al. 2012], mas que ainda não dá suporte à modelagem de *workflow*. Sua arquitetura é apresentada na Figura 2. O usuário especifica as configurações do sistema usando a interface gráfica. O modelo especificado passa pelo interpretador de modelos internos e é entregue ao motor de simulação, baseado em filas de eventos discretos, o qual devolve os resultados para a interface gráfica. Este trabalho estende a forma de modelar a carga na interface gráfica, e da entrega dessa carga para o motor através do modelo simulável. Na Figura 2 também está denotado quais partes da arquitetura foram consideradas na implementação da modelagem de tarefas e sua execução no iSPD.

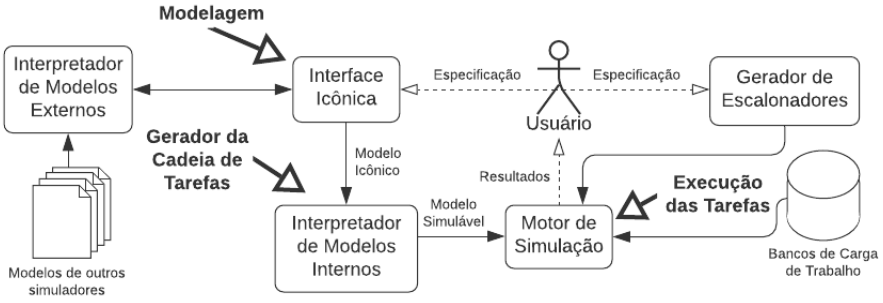

Figura 2. Diagrama conceitual do iSPD com indicação dos pontos de alteração.

Após especificado, durante a interpretação do modelo, é preparada uma cadeia de tarefas, através do algoritmo da Figura 3, para ser processada pelo motor.

```
1: Enquanto houver vértices não verificados no DAG
2:     Gera tarefa a partir de um vértice
3:     Para cada um de seus vértices vizinhos
4:         Encadeia a tarefa gerada adequadamente
5: Retorna as tarefas inicias da cadeia
```

Figura 3. Algoritmo para construir a cadeia de tarefas a partir do DAG

4. Conclusões e Resultados Esperados

O modelo proposto neste trabalho leva em consideração tanto a facilidade de modelagem através de DAG, quanto o detalhamento das interdependências especificado pelo LOTOS, além de possibilitar modelagem simples de *workflows* complexos através das expansões. Isso se mostra útil para simulação de testes de desempenho e habilita o simulador para testagem de escalonadores cientes de dependência.

Em seguida será configurada uma nuvem no sistema real para testes de desempenho utilizando *workflows* como o LIGO, e seus resultados serão comparados com testes equivalentes no simulador.

5. Agradecimentos

À Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP) pelo fomento do projeto através da bolsa de iniciação científica.

Referências

Bolognesi, T. and Brinksma, E. (1987). Introduction to the iso specification language lotos. *Computer Networks and ISDN systems*, 14(1):25–59.

Calheiros, R. N., Ranjan, R., Beloglazov, A., Rose, C. A. D., and Buyya, R. (2011). Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Software: Practice and Experience*, 41(1):23–50.

Manacero, A., Lobato, R. S., Oliveira, P. H., Garcia, M. A., Guerra, A. I., Aoqui, V., Menezes, D., and Da Silva, D. T. (2012). iSPD: an iconic-based modeling simulator for distributed grids. In *Proceedings of the 45th Annual Simulation Symposium*, page 5. Society for Computer Simulation International.

Ramakrishnan, A., Singh, G., Zhao, H., Deelman, E., Sakellariou, R., Vahi, K., Blackburn, K., Meyers, D., and Samidi, M. (2007). Scheduling data-intensiveworkflows onto storage-constrained distributed resources. In *Seventh IEEE International Symposium on Cluster Computing and the Grid (CCGrid'07)*, pages 401–409. IEEE.

Satish, N. R., Ravindran, K., and Keutzer, K. (2008). Scheduling task dependence graphs with variable task execution times onto heterogeneous multiprocessors. In *Proceedings of the 8th ACM international conference on Embedded software*, pages 149–158. ACM.

Tick, J. (2006). Workflow model representation concepts. In *Proceedings of 7th International Symposium of Hungarian Researchers on Computational Intelligence, HUCI*, pages 24–25.