Uma avaliação da Distribuição de Atividades Estática e Dinâmica em Ambientes Paralelos usando o Hydra*

Vítor Silva[1], Fernando Chirigati[1], Eduardo Ogasawara[1, 2], Jonas Dias[1], Daniel de Oliveira[1], Fábio Porto[3], Patrick Valduriez[4], Marta Mattoso[1]

[1] Programa de Engenharia de Sistemas e Computação – COPPE / UFRJ

[2] Centro Federal de Educação Tecnológica Celso Suckow da Fonseca – CEFET/RJ

[3] LNCC – Laboratório Nacional de Computação Científica

[4] INRIA & LIRMM

{silva,fernando_seabra,ogasawara,jonasdias,danielc,marta}@cos.ufrj.br,
fporto@lncc.br, Patrick.Valduriez@inria.fr

Abstract. *Scientific Workflows are used as a basic tool to design and execute scientific experiments on different computational environments. These workflows can be computational and data intensive, requiring high performance computing. The way in which workflow activities are parallelized and distributed over these environments affects the overall performance of the workflow. This work evaluates two different strategies for activity distribution (static and dynamic) using Hydra middleware integrated with VisTrails. Our experiments show that using the right strategy decreases elapsed time for activity distribution in 30%.*

Resumo. *Workflows Científicos são usados como uma abstração básica para estruturação e execução de experimentos científicos em diferentes ambientes computacionais. Estes workflows podem ser intensivos tanto computacionalmente quanto na manipulação de dados, exigindo computação de alto desempenho. A paralelização e distribuição adequada de atividades nestes ambientes influenciam o desempenho do workflow científico como um todo. Este trabalho avalia duas estratégias de distribuição (estática e dinâmica) de atividades utilizando o middleware Hydra integrado ao VisTrails. Nossos experimentos observaram que usar a estratégia adequada diminui a duração para distribuição de atividades em 30%.*

1. Introdução

Muitos experimentos científicos são baseados em simulações computacionais complexas que consomem e produzem conjuntos de dados extensos e utilizam grandes quantidades de recursos computacionais. Conforme a complexidade do experimento cresce, executar tais simulações torna-se um desafio. Portanto, é comum a utilização de workflows científicos com o objetivo de auxiliar os cientistas na gerência de recursos envolvidos em simulações computacionais de larga escala (Mattoso et al. 2010). Um workflow pode ser definido como um modelo de um processo, que consiste em uma série de atividades e suas dependências (Aalst and Hee 2002). Um workflow científico estrutura o processamento de uma simulação científica como um grafo de atividades, nos quais os nós correspondem a atividades que processam dados e as arestas representam os fluxos de dados entre eles. Atividades de workflow estão associadas a programas científicos que preparam, processam e analisam dados.

* Esse trabalho foi parcialmente financiado pela CAPES, CNPq e INRIA (projeto SARAVA, equipe associada).

Sistemas de Gerência de Workflows Científicos (SGWfC) (Deelman et al. 2009) são *softwares* que apoiam a definição, execução e monitoramento de workflows científicos. Existem diversos SGWfC, tais como VisTrails (Callahan et al. 2006) e Swift (Zhao et al. 2007). Cada um deles tem sua própria linguagem (Deelman et al. 2009) e foca em diferentes aspectos, tais como execução paralela, suporte semântico, características específicas de domínio e proveniência dos dados.

Em um workflow científico, uma mesma atividade pode ser executada repetidas vezes variando-se os valores de seus parâmetros. Assim, a distribuição de atividades de um workflow na execução paralela se apresenta como uma característica importante para os SGWfC em experimentos de longa duração. Essa distribuição em ambientes de Processamento de Alto Desempenho (PAD) é essencial para obter os resultados de experimentos em larga escala em um tempo hábil. Existem soluções que visam a apoiar essa distribuição, tais como o Swift, o MapReduce (Wang et al. 2009) e o Hydra (Ogasawara et al. 2009).

Nas soluções de distribuição de atividades, um fator que precisa ser levado em consideração é a estratégia adotada. Diante da grande quantidade de valores de parâmetros que precisam ser processados pelas atividades envolvidas na execução de um workflow científico, a estratégia de distribuição pode afetar o desempenho da solução. Existem basicamente dois extremos no que diz respeito a estratégias de distribuição. A primeira estratégia é a distribuição estática, onde cada *core* recebe o conjunto de instâncias de atividades que ele deve processar antes de começar a execução. A segunda estratégia é a distribuição dinâmica, onde cada *core* requisita uma nova instância de atividade conforme sua disponibilidade. Assim, o objetivo deste trabalho é avaliar a variação de desempenho na utilização dessas duas estratégias.

A avaliação foi realizada utilizando o VisTrails com o Hydra. O Hydra foi escolhido por apoiar diferentes estratégias de distribuição e poder ser utilizado em diversos SGWfC. O Hydra fornece um conjunto de componentes que podem ser incluídos na especificação do workflow de um SGWfC para controlar a distribuição de atividades em ambientes de PAD. O Hydra também permite a configuração da estratégia de distribuição e a coleta de proveniência distribuída.

Este artigo está organizado em cinco seções, além desta introdução. A Seção 2 apresenta a arquitetura do middleware Hydra. Os métodos de distribuição de atividades em ambientes de PAD são discutidos na Seção 3. A Seção 4 apresenta os resultados experimentais. Os trabalhos correlatos são detalhados na Seção 5 e a Seção 6 conclui o artigo.

2. Middleware Hydra

O Hydra (Ogasawara et al. 2009) é um middleware capaz de distribuir, controlar e monitorar a execução paralela das atividades de um workflow científico em ambientes de PAD. Ele é desenvolvido em Java 1.6 e utiliza MPJ (http://mpj-express.org) para a comunicação paralela. Para o armazenamento de dados de proveniência, o PostgreSQL 8.4 (http://www.postgresql.org/) é utilizado.

O Hydra oferece ao cientista a distribuição automática de uma determinada atividade em ambientes de PAD, através de poucas alterações na definição do workflow. Pode-se citar como características positivas do Hydra: (i) não associação a uma máquina de execução do SGWfC específica; (ii) simplificação do processo de distribuição de atividades em PAD; e (iii) coleta de proveniência distribuída.

Um dos tipos de paralelismo suportado pelo Hydra é a varredura de parâmetros (Walker and Guiang 2007). Neste modelo de paralelismo, os cientistas exploram o comportamento do seu modelo computacional com diferentes entradas, configuradas no workflow. A varredura de parâmetros é muito comum e pode representar uma barreira para os cientistas controlarem e monitorarem a execução do experimento, sendo então importante fornecer apoio para tal paralelismo. Para explicar a varredura de parâmetros, apresentaremos uma formalização de workflow científico adaptada de Meyer et al. (2007).

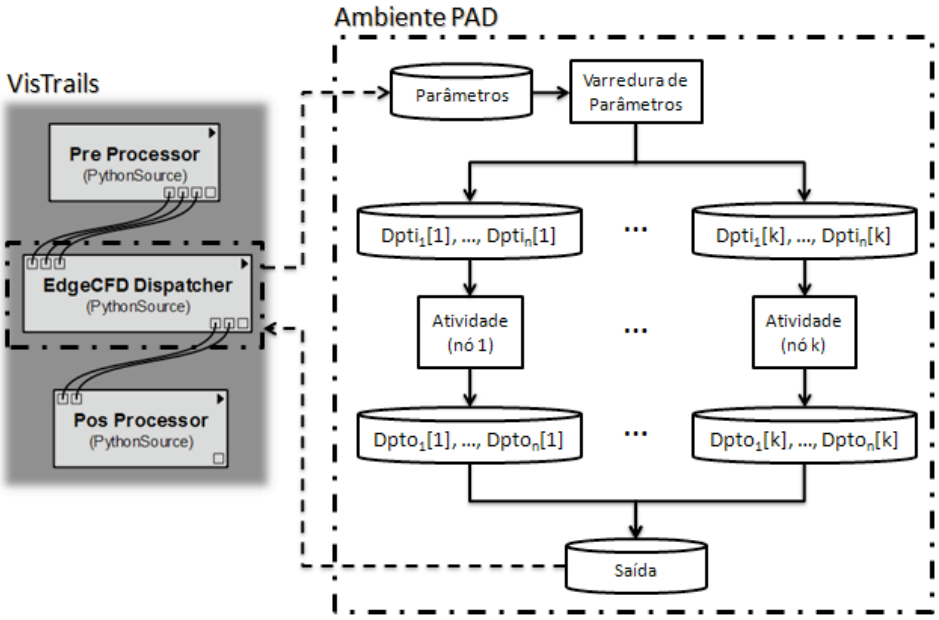


Figura 1. Varredura de parâmetros realizada pelo Hydra

Um workflow científico *Wf* é representado por uma tripla (*A*, *PtI*, *PtO*), onde: *A* é um conjunto $\{a_1, a_2, ..., a_n\}$ de atividades que constituem *Wf*; *PtI* é o conjunto $\{pti_1, pti_2, ..., pti_m\}$ de parâmetros de entrada de *Wf*; e *PtO* é o conjunto $\{pto_1, pto_2, ..., pto_r\}$ de parâmetros de saída de *Wf*. Cada parâmetro em *PtI* e em *PtO* pode assumir valores no domínio associado a ele. Assim, $Dpti_j$ representa o domínio de possíveis valores para o parâmetro pti_j, enquanto que $Dpto_j$ representa o domínio de possíveis valores para o parâmetro pto_j.

A varredura de parâmetros efetuada pelo Hydra é caracterizada por diversas e simultâneas execuções paralelas de uma atividade a_j do workflow, em que cada execução usa diferentes valores para os parâmetros de *PtI*. Por exemplo, suponha que a atividade a_j do workflow *Wf* possua parâmetros de entrada pti_1 e pti_2, e que os domínios de possíveis valores para esses parâmetros sejam $Dpti_1 = \{x, x'\}$ e $Dpti_2 = \{y, y'\}$. Uma instância da atividade a_j de *Wf* pode consumir os valores *x* e *y* para os parâmetros pti_1 e pti_2, respectivamente, enquanto outra instância da mesma atividade pode consumir os valores *x'* e *y'*. O paralelismo para essa varredura de parâmetros é realizada alocando-se diferentes configurações de valores para os parâmetros de *PtI*, em que cada configuração está associada a um diferente *core* do ambiente de PAD. A Figura 1 apresenta o esquema desse paralelismo aplicado a um workflow científico desenvolvido no SGWfC VisTrails, em que a atividade a_j é representada pela atividade EdgeCFD Dispatcher, onde a mesma realiza a varredura de parâmetros.

3. Distribuição estática e distribuição dinâmica no Hydra

Dado um determinado cenário de paralelização de uma atividade por meio de varredura de parâmetros, pode-se estabelecer um conjunto de instâncias desta atividade. Uma instância da atividade é uma execução da atividade que consome uma determinada combinação de valores de parâmetros. As instâncias podem ser distribuídas dentre os *cores* computacionais que irão processá-las num ambiente de PAD, tal como um cluster. No Hydra, o conjunto de instâncias da atividade paralelizada pode ser distribuído seguindo as estratégias de distribuição estática e dinâmica.

Na estratégia estática, as instâncias da atividade de um dado workflow são pré-alocadas para cada *core* antes de serem consumidas. Ou seja, a distribuição do número de instâncias por *core* é feita *a priori*. A Figura 2(a) representa um esquema da distribuição estática de atividades. Nesta figura, cada instância da atividade A (as instâncias estão representadas pelos quadrados *a*) é referente a um mesmo programa com um conjunto diferente de valores de parâmetros. No início do processo paralelo, o escalonador atribui para cada *core* um número fixo de quatro instâncias da atividade A para serem processadas. O escalonador apresentado na figura é o componente do Hydra responsável pela distribuição. Na estratégia de distribuição estática, o escalonador do Hydra utiliza a abordagem *gather* e *scatter* do MPJ para transmitir as tarefas aos *cores*.

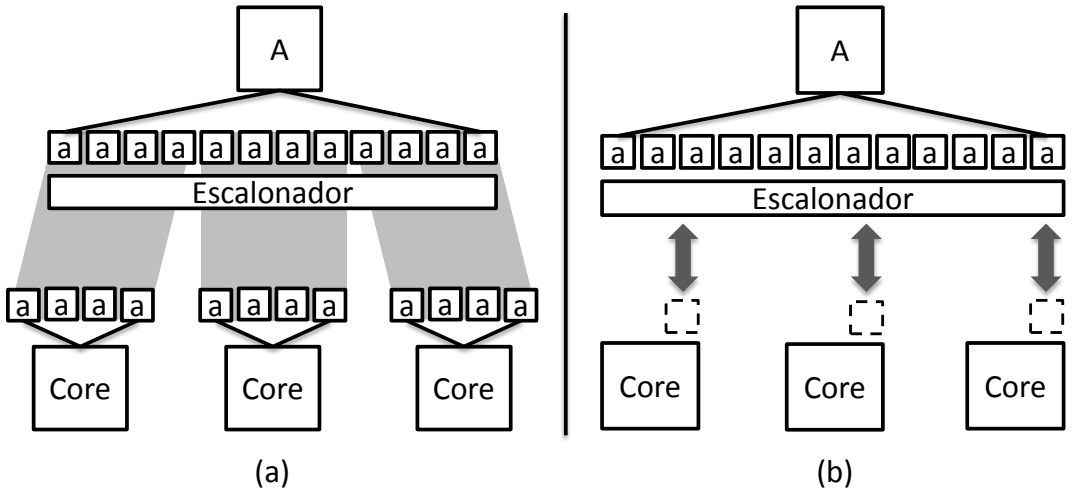
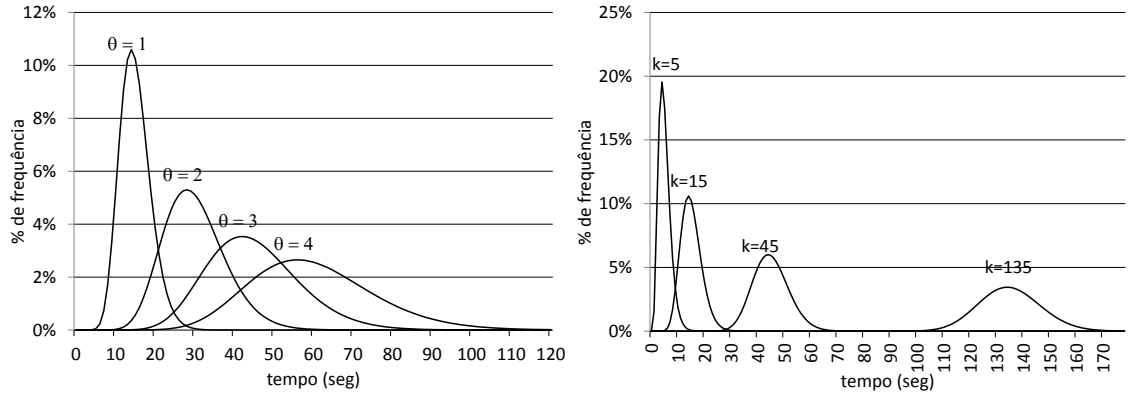


Figura 2. Estratégias de distribuição estática (a) e dinâmica (b)

Por outro lado, na estratégia dinâmica, cada instância da atividade é alocada para um *core* conforme este esteja disponível e requisite uma instância para processar. Ou seja, as instâncias da atividade permanecem em uma fila única e cada núcleo obtém e processa apenas uma instância da atividade por vez. Na Figura 2(b) é apresentado um esquema da distribuição dinâmica, onde cada *core* requisita ao escalonador uma instância *a* da atividade A por vez. Neste caso, o escalonador do Hydra precisa manter uma *thread* que aguarda requisições feitas pelos *cores* através de primitivas de comunicação do tipo *send* e *receive* do MPJ.

4. Avaliação Experimental

Nesta seção, apresentamos uma comparação de desempenho entre as distribuições estática e dinâmica de atividades em cenários de varredura de parâmetros. Os estudos foram conduzidos utilizando o Hydra como middleware de distribuição de instâncias de atividades em um cluster SGI Altix ICE 8200 de 256 cores (32 nós, cada nó contendo dois processadores quad-core Xeon).



(a) (b)

Figura 3. Duração de uma atividade seguindo uma distribuição Gamma com k = 15 fixo e θ variando de 1 a 4 (a); Duração de uma atividade seguindo uma distribuição Gamma, com θ = 1 fixo e k variando em 5, 15, 45 e 135 (b).

Inicialmente, realizamos um estudo em que as atividades possuíam duração fixa de um segundo. Constatou-se que a estratégia estática obteve um desempenho superior em aproximadamente 30%, quando comparada à estratégia dinâmica.

Em outro estudo, utilizamos dados sintéticos para avaliar atividades de duração variada. Consideramos oito tipos de atividades diferentes, cada uma delas seguindo uma distribuição Gamma (Freedman et al. 2007) específica, conforme apresentado nos cenários (a) e (b), da Figura 3. A distribuição Gamma, representada por $\Gamma(k, \theta)$ nesses cenários, tem k como parâmetro de forma e θ como parâmetro de escala ($k, \theta > 0$). Cabe ressaltar que a distribuição Gamma é adequada, por frequentemente ser utilizada como modelo de probabilidade para tempo de espera (Weisstein 2011). No cenário (a), temos quatro tipos de atividades, todas com o mesmo k fixo de 15 e cada uma delas com um θ diferente (1, 2, 3 e 4). Já no cenário (b), temos quatro tipos de atividades, todas com o θ fixado em 1 e cada uma delas com um k diferente (5, 15, 45 e 135).

O estudo mediu a duração da distribuição das atividades para cada tipo de atividade, ou seja, o tempo necessário para processar todas as instâncias de um determinado tipo. Em cada uma das medições foi realizada uma varredura de parâmetros de 1024 combinações de valores de parâmetros numa distribuição de 64 cores. A Figura 4 apresenta os resultados para os dois cenários mostrando o desempenho das distribuições estática e dinâmica. A estratégia dinâmica mostrou-se superior em todas estas situações, especialmente onde a variância da distribuição é maior, *i.e.*, o valor de θ é maior. No cenário (b), de variância fixa (θ fixo), observamos que ambas as estratégias escalam de maneira muito similar, apesar de a estratégia dinâmica apresentar melhores resultados.

Para se tentar justificar esta diferença, fizemos uma segunda análise verificando a porcentagem de recursos ociosos durante a execução. A Figura 5(a) apresenta os resultados de ambas as estratégias. Os resultados explicitam a razão da superioridade da estratégia dinâmica. Ela minimiza a ociosidade dos recursos, pois funciona com um mecanismo de fila única enquanto a estratégia estática funciona com o mecanismo de uma fila para cada *core*, o que pode causar um desbalanceamento. Com menos ociosidade, a duração da execução paralela tende a ser menor.

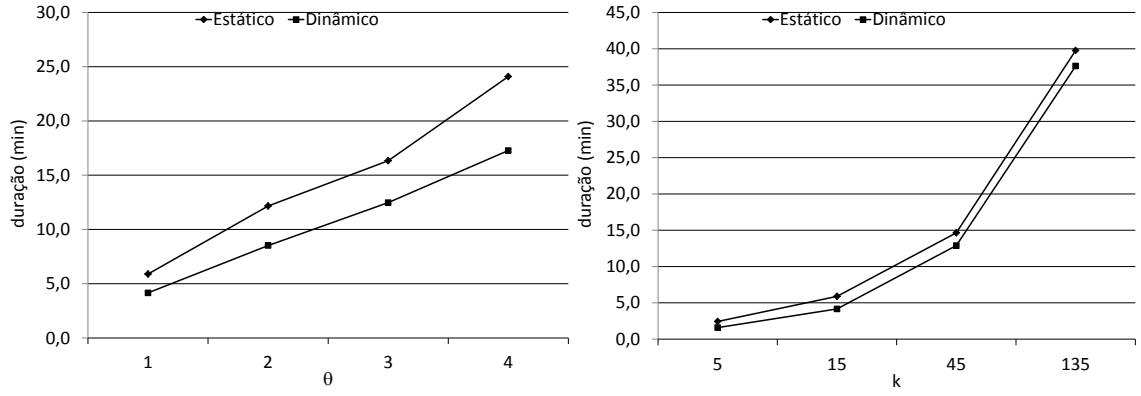


Figura 4. Duração de distribuição de atividades segundo distribuição Gamma com k = 15 e θ variando de 1 a 4 (a); Duração de distribuição de atividades segundo distribuição Gamma com θ = 1 fixo e k variando (5, 15, 45 e 135) (b).

Uma última análise apresentada na Figura 5(b) mostra a melhora do desempenho com o aumento do número de recursos. Pode-se observar que o comportamento da estratégia dinâmica permanece melhor, embora a diferença de desempenho para a estratégia estática diminua com o aumento da quantidade de recursos. Entretanto, essa diferença de desempenho também está associada à quantidade de instâncias de atividades para serem consumidas.

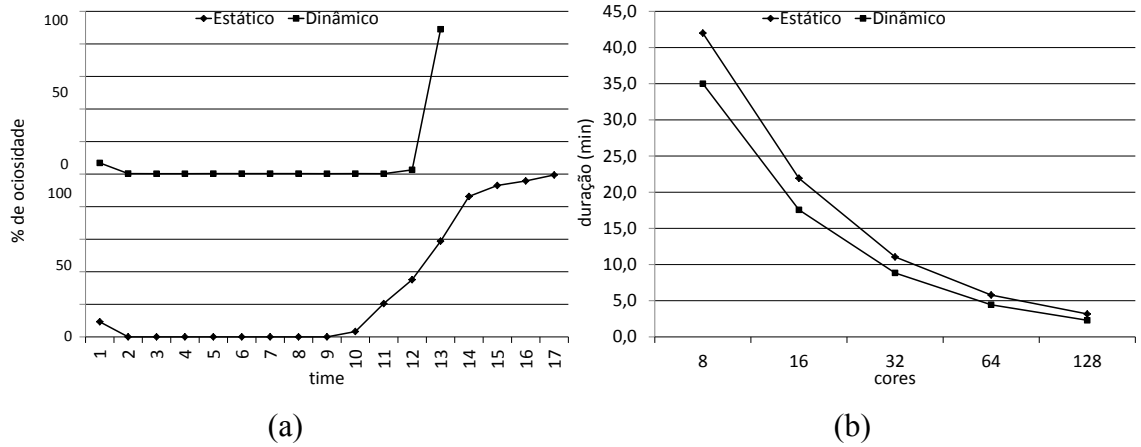


Figura 5. % de ociosidade de cores para distribuição dinâmica e estática (a); Comparação de desempenho variando-se o número de core (b).

5. Trabalhos Relacionados

As soluções atuais voltadas ao paralelismo em workflows científicos, como o SGWfC Swift, são capazes de executar atividades paralelas em diferentes ambientes computacionais como clusters, grades computacionais e nuvens. Em geral, tais soluções podem ser acopladas a escalonadores existentes como o Condor (Thain et al. 2002). Embora o Condor seja capaz de distribuir as atividades de maneira dinâmica dentre os *cores*, ele demora cerca de meio segundo para disparar cada atividade (Raicu et al. 2008). Em um cenário com centenas ou milhares de instâncias de atividade, essa taxa de distribuição é indesejada. Além disso, a política de filas do Condor pode limitar o número de atividades submetidas por usuário. Mais especificamente no caso do Swift, o mesmo pode ser também acoplado ao Falkon (Raicu et al. 2007). O Falkon é um

escalonador que pode ser usado em substituição ao Condor, mas a sua estratégia de distribuição adotada não é divulgada pela equipe de desenvolvimento.

O Nimrod/K é um conjunto de complementos para o SGWfC Kepler cujo objetivo é orquestrar a execução do workflow realizando a varredura de parâmetros em clusters, grades e nuvens. Entretanto, o Nimrod/K só funciona com o Kepler e limita-se a apoiar a varredura de parâmetros por meio de distribuição estática das instâncias de atividades do workflow. Uma solução baseada no paradigma MapReduce, como o Hadoop (http://hadoop.apache.org), integrada a um SGWfC tradicional, como o Kepler, também pode realizar a distribuição de atividades de workflow. O Hadoop é capaz de mapear diferentes execuções de atividades do workflow em ambientes distribuídos como clusters, grades e nuvens. Após a execução, os resultados são agrupados. Existe integração do Kepler com o Hadoop, mas o workflow fica dependente do SGWfC e da capacidade do cientista em programar as funções de Map e Reduce no Kepler. O Hadoop também realiza somente a distribuição estática das instâncias da atividade paralela do workflow.

6. Conclusão

Experimentos científicos em larga escala são caracterizados por cálculos computacionais complexos envolvendo simulações. A utilização de workflows científicos para a concepção, execução e controle destes experimentos é muito comum e a longa duração das atividades dos workflows implica na necessidade de utilizar a computação paralela em ambientes de PAD, a fim de diminuir o tempo de execução dos workflows científicos. A varredura de parâmetros de uma determinada atividade do workflow pode ser paralelizada de modo a se obter os resultados em tempo hábil. Cada instância da atividade pode ser alocada para executar em paralelo em diferentes *cores*. Porém, a estratégia de distribuição pode influenciar o desempenho total da execução. As soluções atuais para processamento paralelo em SGWfC usam escalonadores, como o Condor, para distribuir uma determinada atividade. Nesses escalonadores, porém, não é possível configurar a estratégia de distribuição a ser utilizada. Dessa forma, este trabalho apresenta um estudo comparativo das estratégias estática e dinâmica utilizando o Hydra e o SGWfC VisTrails, com o intuito de apoiar a tomada de decisões em experimentos que demandem paralelismo em workflows.

O estudo mostrou que a estratégia estática apresenta desempenho superior quando a atividade distribuída possui uma curta duração. Em outros casos, porém, a estratégia dinâmica apresenta um desempenho superior, pois ela minimiza a ociosidade dos *cores*. A abordagem dinâmica também é menos sensível a variância do tempo de execução da atividade. Essa superioridade ocorre em função do mecanismo de fila única utilizado pela estratégia dinâmica que evita o desbalanceamento presente no mecanismo de filas múltiplas na distribuição estática. Mesmo com o aumento do número de *cores* presentes na execução, a estratégia dinâmica manteve o melhor desempenho. Esses resultados apontam a importância na escolha da estratégia de distribuição, embora a maioria das soluções atuais como o Swift, Nimrod/K e o Hadoop desconsiderem tal opção.

Agradecimentos

Os autores agradecem ao NACAD/UFRJ pelo uso da infraestrutura computacional.

Referências

Aalst, W. van der, Hee, K. van, (2002), *Workflow Management: Models, Methods, and Systems*. 1 ed. The MIT Press.

Callahan, S. P., Freire, J., Santos, E., Scheidegger, C. E., Silva, C. T., Vo, H. T., (2006), "VisTrails: visualization meets data management". In: *Proc. SIGMOD*, p. 745-747, Chicago, Illinois, USA.

Deelman, E., Gannon, D., Shields, M., Taylor, I., (2009), "Workflows and e-Science: An overview of workflow system features and capabilities", *Future Generation Computer Systems*, v. 25, n. 5, p. 528-540.

Freedman, D., Pisani, R., Purves, R., (2007), *Statistics, 4th Edition*. 4 ed. W. W. Norton.

Mattoso, M., Werner, C., Travassos, G. H., Braganholo, V., Murta, L., Ogasawara, E., Oliveira, D., Cruz, S. M. S. da, Martinho, W., (2010), "Towards Supporting the Life Cycle of Large Scale Scientific Experiments", *International Journal of Business Process Integration and Management*, v. 5, n. 1, p. 79–92.

Meyer, L., Scheftner, D., Vöckler, J., Mattoso, M., Wilde, M., Foster, I., (2007), "An Opportunistic Algorithm for Scheduling Workflows on Grids", *VECPAR 2006*, 1 ed, p. 1-12.

Ogasawara, E., Oliveira, D., Chirigati, F., Barbosa, C. E., Elias, R., Braganholo, V., Coutinho, A., Mattoso, M., (2009), "Exploring many task computing in scientific workflows". In: *MTAGS 09*, p. 1-10, Portland, Oregon.

Raicu, I., Foster, I. T., Yong Zhao, (2008), "Many-task computing for grids and supercomputers". In: *Workshop on Many-Task Computing on Grids and Supercomputers*, p. 1-11, Austin, Texas.

Raicu, I., Zhao, Y., Dumitrescu, C., Foster, I., Wilde, M., (2007), "Falkon: a Fast and Light-weight tasK executiON framework". In: *SC07*, p. 1-12, Reno, Nevada.

Thain, D., Tannenbaum, T., Livny, M., (2002), "Condor and the Grid", *Grid Computing: Making the Global Infrastructure a Reality. John Wiley & Sons Inc*

Walker, E., Guiang, C., (2007), "Challenges in executing large parameter sweep studies across widely distributed computing environments". In: *Workshop on Challenges of large applications in distributed environments*, p. 11-18, Monterey, California, USA.

Wang, J., Crawl, D., Altintas, I., (2009), "Kepler + Hadoop: a general architecture facilitating data-intensive applications in scientific workflow systems". In: *Proc. of the 4th Workshop on Workflows in Support of Large-Scale Science*, p. 1-8, Portland, Oregon.

Weisstein, E. W., (2011), *Gamma Distribution*, Text, http://mathworld.wolfram.com/GammaDistribution.html.

Zhao, Y., Hategan, M., Clifford, B., Foster, I., von Laszewski, G., Nefedova, V., Raicu, I., Stef-Praun, T., Wilde, M., (2007), "Swift: Fast, Reliable, Loosely Coupled Parallel Computation". In: *Proc. of the 3rd IEEE World Congress on Services*, p. 206, 199, Salt Lake City, USA.