SimiFlow: Uma Arquitetura para Agrupamento de *Workflows* por Similaridade

Vítor Silva[1], Fernando Chirigati[1], Kely Maia[2], Eduardo Ogasawara[1], Daniel de Oliveira[1], Vanessa Braganholo[2], Leonardo Murta[3], Marta Mattoso[1]

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

[2]Departamento de Ciência da Computação – IM / UFRJ

[3]Instituto de Computação – Universidade Federal Fluminense

{silva,fernando_seabra,ogasawara,danielc,marta}@cos.ufrj.br,
{kely.maia,braganholo}@dcc.ufrj.br, leomurta@ic.uff.br

Abstract. *Scientists have been using scientific workflows to support scientific experiments. However, the Scientific Workflow Management Systems present some limitation on workflow composition. Experiment Lines, which are a novel approach to deal with these limitations, allow the representation and systematic composition of the experiment. Nevertheless, there are many scientific workflows already modeled that can leverage the construction of experiment lines via the identification of scientific workflows clusters that are created according to similarity. This paper proposes SimiFlow, an architecture for similarity-based comparison and clustering to build experiment lines following a bottom-up approach.*

Resumo. *Workflows científicos vêm sendo utilizados no apoio aos experimentos científicos. Workflows em um mesmo experimento normalmente apresentam pequenas variações, que são difíceis de gerenciar utilizando os Sistemas de Gerência de Workflows Científicos tradicionais. O conceito de Linhas de Experimentos foi definido para suprir esta lacuna, pois possibilitam a representação de um experimento e promovem a composição de workflows científicos de maneira sistemática. No entanto, as linhas de experimentos ainda não resolvem o problema de sistematização de workflows pré-existentes. Este artigo apresenta o SimiFlow, uma arquitetura para comparação e agrupamento de workflows pré-existentes por similaridade visando a construção de linhas de experimentos por meio de abordagem ascendente.*

1. Introdução

Experimentos científicos vêm recebendo forte apoio computacional nos últimos anos. Muitos destes experimentos necessitam processar um grande volume de dados, o que se torna inviável sem apoio computacional. Em vários cenários, cientistas executam uma grande quantidade de simulações a fim de avaliar suas hipóteses. Estas simulações são fortemente dependentes do uso de recursos computacionais, muitas vezes complexos. Os *workflows* científicos são uma abstração que permite a especificação destes experimentos, de maneira estruturada, através de um fluxo de programas, serviços e dados para produzir um resultado final (Deelman et al. 2009).

Estes *workflows* científicos precisam ser representados em diversos níveis de abstração. De acordo com Cavalcanti *et al.* (2005), um *workflow* é chamado de abstrato quando as atividades que o definem não estão ligadas a recursos computacionais, como programas ou serviços. Já um *workflow* concreto tem suas atividades mapeadas

diretamente para recursos computacionais. *Workflows* concretos são usualmente modelados em Sistemas de Gerência de *Workflows* Científicos (SGWfC). Entretanto, esses sistemas não oferecem meios para representações abstratas, o que motiva a existência de pesquisa em ferramentas e técnicas para esse fim (Mattoso et al. 2010).

De forma a preencher esta lacuna, o conceito de linhas de experimentos (Ogasawara et al. 2009) foi criado a fim de apoiar a composição de *workflows* científicos voltados a experimentos *in silico*. O conceito de linhas de experimentos tem como objetivo minimizar a complexidade na composição de experimentos ao definir famílias de experimentos com características em comum. As linhas de experimentos atuam em um nível de abstração elevado e podem ser vistas como um meta-experimento para uma determinada área do conhecimento, que representa o que existe em todos os experimentos dessa área, o que é opcional e o que é variante. A partir das linhas de experimentos, é possível derivar *workflows* concretos por meio da seleção das opções e dos pontos de variação disponíveis, a fim de atender a um experimento específico.

Existem duas maneiras distintas de se criar linhas de experimentos: do nível abstrato para o concreto (descendente) ou do nível concreto para o abstrato (ascendente). A primeira abordagem pode ser utilizada quando os cientistas estão iniciando a concepção de um novo experimento e ainda não possuem *workflows* concretos disponíveis que possam ser reutilizados. Assim, as linhas de experimentos são criadas diretamente pelos cientistas e, a partir delas, são derivados *workflows* concretos que apoiam a execução de experimentos específicos.

Entretanto, este não é o cenário de muitos laboratórios de pesquisa, que possuem *workflows* pré-existentes (Mattoso et al. 2010). De fato, esses *workflows* são concretos na grande maioria das vezes. Ou seja, já existe uma gama de *workflows* concretos modelados e que, muitas vezes, são representações de um mesmo *workflow* abstrato, sem que o cientista tenha conhecimento deste fato. Desta forma, uma vez sendo possível comparar os *workflows* concretos já modelados, é possível organizar melhor o conhecimento do experimento, permitindo uma abordagem ascendente para construções de linhas de experimentos. Para seguir essa abordagem ascendente, são necessárias primitivas que indiquem similaridade entre os *workflows*, bem como primitivas para agrupá-los (*clustering*).

Este artigo apresenta o SimiFlow, uma arquitetura extensível para a comparação de *workflows* por similaridade e agrupamento de *workflows*, visando a construção de linhas de experimentos de acordo com uma abordagem ascendente. O foco desse trabalho, contudo, é o processo de comparação e agrupamento, e não a geração das linhas. Essa arquitetura foi implementada na ferramenta GExpLine (GExp 2009), responsável pela gerência das linhas de experimentos.

Este artigo está organizado em quatro seções, além desta introdução. A Seção 2 apresenta a arquitetura da SimiFlow. A Seção 3 discute resultados preliminares obtidos com a SimiFlow. Os trabalhos correlatos são detalhados na Seção 4 e a Seção 5 conclui.

2. SimiFlow

Esta seção apresenta a arquitetura da SimiFlow, desenvolvida com o objetivo de comparar *workflows* por similaridade e agrupá-los. Sendo assim, está subdividida em quatro subseções. A subseção 2.1 apresenta a arquitetura desenvolvida. As subseções 2.2, 2.3 e 2.4 expõem, respectivamente, a importação de *workflows*, a comparação de *workflows* por similaridade e o agrupamento de *workflows*.

2.1 Arquitetura

A arquitetura SimiFlow foi projetada com o objetivo de analisar e agrupar um conjunto de *workflows* científicos com base na similaridade calculada de cada elemento do conjunto em relação a todos os outros. A partir da similaridade produzida, a arquitetura é capaz de gerar agrupamentos de *workflows*. Estes agrupamentos têm como membros *workflows* que apresentam características comuns entre si, representando uma família de *workflows*. Como as linhas de experimentos representam famílias de *workflows*, podemos utilizar a arquitetura SimiFlow para gerar linhas de experimentos baseadas em um determinado conjunto de *workflows* executáveis.

Devido à existência de diversas análises de similaridade, é necessário que a arquitetura seja extensível, de modo a propiciar o desenvolvimento e a utilização de diferentes algoritmos ou métodos para as operações de comparação por similaridade e de agrupamento. Neste sentido, a arquitetura da SimiFlow foi concebida por meio da aplicação do padrão *Strategy* (Larman 2004), que é pautada na troca dinâmica de algoritmos desenvolvidos para uma aplicação. Logo, é possível definir um grupo ou família de algoritmos, com o encapsulamento em objetos.

De forma a apoiar tal finalidade, foram desenvolvidos três cartuchos para a arquitetura SimiFlow: o cartucho de importação de *workflows*, o cartucho de comparação por similaridade e o cartucho de agrupamento. O primeiro cartucho é responsável pela obtenção de dados importantes de *workflows* modelados em SGWfC e o seu posterior armazenamento na base de dados da GExpLine. O cartucho de comparação por similaridade, por sua vez, compara *workflows* dois a dois, tendo como resultado a similaridade, que é salva na base de dados da ferramenta GExpLine. O último cartucho trabalha com o agrupamento de *workflows* propriamente dito. A Figura 1(a) mostra a arquitetura proposta e seus respectivos cartuchos.

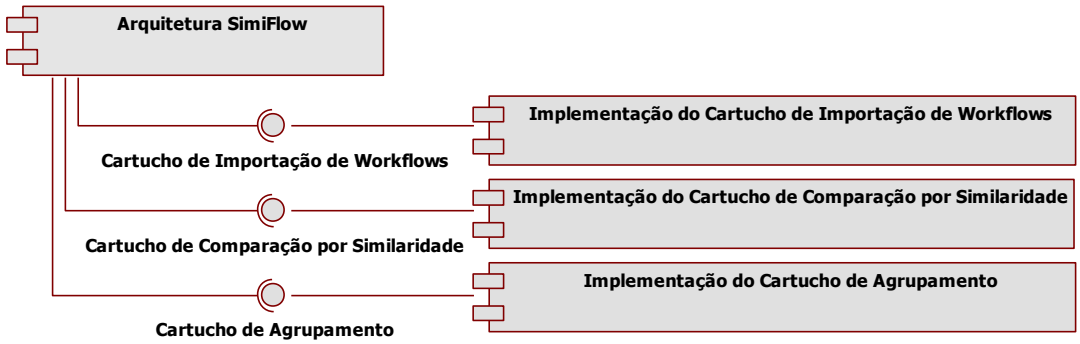

Figura 1. A arquitetura SimiFlow

2.2 Importação de *Workflows*

O cartucho importação de *workflows* apoia o armazenamento das propriedades de *workflows* originados dos SGWfC Taverna (Oinn et al. 2004), VisTrails (Callahan et al. 2006) e Kepler (Altintas et al. 2004) na base de dados criada para a ferramenta GExpLine. Estes SGWfC geram arquivos XML de forma nativa para representar os *workflows* concretos modelados. Tais arquivos XML são importados e analisados pelos componentes do SimiFlow, que são capazes de mapear as informações presentes nos arquivos XML para a base de dados GExpLine.

Para uma avaliação dos algoritmos de comparação por similaridade e de agrupamento, é importante trabalhar com uma base de dados que tenha um número

considerável de *workflows*. Para isto, a população da base de dados da GExpLine com *workflows* científicos foi obtida no sítio do *myExperiment* (Goble e Roure 2007) de modo automático e incremental. O *myExperiment* é um repositório de *workflows* provenientes do SGWfC Taverna, com mais de 600 *workflows* cadastrados, sendo 459 *workflows* públicos (dados obtidos no dia 24/11/2009).

2.3 Comparação por Similaridade

O primeiro algoritmo utilizado no SimiFlow para calcular a similaridade entre dois *workflows* é uma variação do algoritmo definido por Seo *et al.* (2007). Neste algoritmo, analisa-se a semelhança entre *workflows* a partir do nome, das portas de entrada e saída existentes nas atividades e do tipo de relacionamento existente entre duas atividades. Entretanto, o algoritmo de Seo *et al.* (2007) não considera outras propriedades dos elementos do *workflow* para o cálculo de similaridade, como, por exemplo, a existência de subworkflows e avaliação dos tipos de portas das atividades.

A similaridade é representada no algoritmo pela variável *Similarity*, que é inicializada com o valor igual a 1. Para cada componente de um *workflow*, seja ele uma atividade, porta, relacionamento ou subworkflow, há uma análise de suas propriedades específicas, avaliando se tais parâmetros são iguais entre os *workflows*. Caso sejam diferentes, um valor de penalização é subtraído do valor presente na variável *Similarity*, indicando que os *workflows* são menos similares. A importância de cada propriedade específica do componente determina o valor da penalização, caso o campo analisado seja diferente entre os *workflows*. Por exemplo, a cada par de atividades analisado, uma comparação entre as portas é realizada. Tal análise avalia alguns critérios: se o tamanho da lista de portas for diferente entre as atividades do par, há uma penalização de 0,1 na similaridade; se a quantidade de portas de entrada é diferente entre as atividades, então a penalização é a diferença do número de portas de entrada em valor absoluto, multiplicado por 0,5. O mesmo vale para as portas de saída. Por último, analisa-se, para cada combinação de portas, se elas têm o mesmo nome e se são do mesmo tipo de *workflow*, e, caso não sejam, há uma penalização de 0,05. Vale ressaltar que os valores apresentados nesse exemplo são configuráveis, e os mesmos são valores padrão utilizados pelo algoritmo.

É importante observar também que, devido à divisão da arquitetura em cartuchos, é possível testar outros algoritmos de comparação por similaridade. Para isso, basta que ele seja implementado como um cartucho e incorporado à arquitetura da SimiFlow.

2.4 Agrupamento

Após o cálculo da similaridade entre os *workflows*, é necessário agrupá-los. Com o intuito de realizar propriamente o processo de agrupamento dos *workflows*, o cartucho de agrupamento realiza o cálculo de similaridade de todos os pares de *workflows* presentes na base. Este resultado, portanto, forma um grafo completo, definido por *G*, com *n* vértices (*n* é o número de *workflows* presentes na base de dados) e *m* arestas ($m = C_2^n$). Após a criação desse grafo, é feito o corte das arestas que possuem uma similaridade menor que o valor de corte de similaridade. O resultado é um grafo em forma de floresta. Os componentes conexos com apenas um vértice são cortados do grafo e, assim, os componentes conexos restantes representam os agrupamentos obtidos.

A Figura 2 apresenta dois exemplos de agrupamento. Na Figura 2(a), a partir dos valores de similaridade, pode-se constatar que há formação de dois grupos, considerando o valor de corte padrão do algoritmo, igual a 0,55, pois as ligações com valores menores que 0,55 são desconsideradas. Assim, os *workflows* 1 e 2 ficam

desconectados dos demais, formando um grupo, e os *workflows* 3 e 4 formam outro grupo. Outro exemplo de agrupamento é apresentado na Figura 2(b). Outros algoritmos podem ser testados sem grandes dificuldades, graças à extensibilidade por meio dos cartuchos da arquitetura.

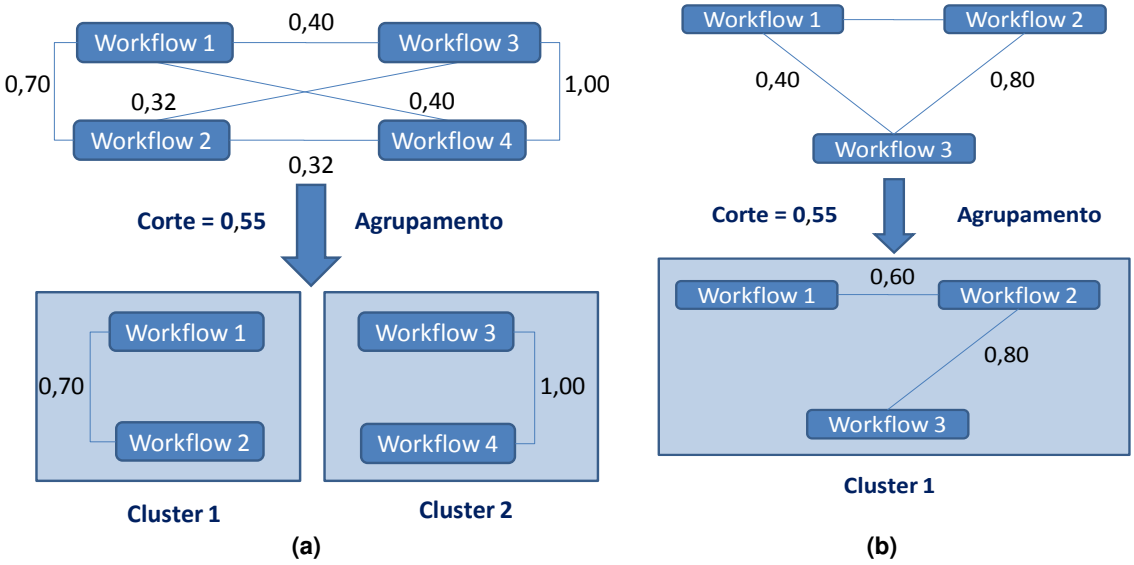

Figura 2. Exemplos de agrupamento de *workflows*

3. Resultados Preliminares

A SimiFlow foi implementada junto à ferramenta GExpLine. Dois estudos preliminares foram realizados com o objetivo de avaliar o quanto a métrica de similaridade está retornando resultados satisfatórios e se o agrupamento dos *workflows* está consistente. O primeiro estudo baseou-se em identificar agrupamentos de *workflows*, enquanto o segundo teve a finalidade de testar a escalabilidade da arquitetura. Os *workflows* utilizados foram os da base importada via *myExperiment*.

O primeiro estudo teve a preocupação de avaliar a capacidade da SimiFlow em realizar os agrupamentos, concentrando-se em analisar e agrupar somente 24 *workflows* do Taverna. A escolha dos 24 *workflows* para esse estudo foi feita de forma manual, buscando-se no sítio do *myExperiment* conjuntos de *workflows* que possuíssem finalidades similares, ou seja, que possuíssem a mesma *tag*. Uma das *tags* utilizadas, por exemplo, foi a *EBI*, para a qual a busca resultou em alguns *workflows*, como o *EBI_ClustalW2* e o *EBI_ClustalW2_phylogentic_tree*. Neste estudo, foi possível observar valores satisfatórios para a similaridade entre os *workflows* e o agrupamento, como, por exemplo, a similaridade acima do valor de corte configurado (0,5) entre os *workflows EBI_Kalign* e *EBI_TCoffee* (0,67), *get Concepts* e *get Cvs* (0,93), e *get Cvs* e *get Evidence Types* (0,93). Todos esses *workflows* são, de fato, conceitualmente semelhantes (as descrições textuais do sítio *myExperiment* comprovam isso). A Tabela 1 revela o resultado de 12 *workflows*, visto que os demais não formaram agrupamentos. A Figura 3 apresenta a semelhança entre os *workflows EBI_Kalign* e *EBI_TCoffee*.

O segundo estudo teve a preocupação em avaliar a escalabilidade do SimiFlow, concentrando-se em trabalhar com um maior número de *workflows*. A máquina utilizada para esse estudo possui um processador Pentium de 2.66GHz, 1GB de memória RAM e sistema operacional Windows XP Professional. Inicialmente, a importação gerou um total de 459 *workflows* na base de dados. O tempo total para a importação foi de, aproximadamente, 1 hora e 20 minutos. Em seguida, foram realizadas as comparações

entre os *workflows*, em um total de 105.111 comparações (C_2^{459}). Para o cálculo dessas comparações, levou-se, aproximadamente, 221 horas (9 dias e 5 horas). Finalmente, a etapa de agrupamento levou por volta de 1 hora para poder realizar os cortes e criar os grupos de *workflows*.

Tabela 1. Agrupamento de *Workflows*

Nome do *Workflow*	Grupo	Nome do *Workflow*	Grupo
Discover_entities	1	Get Concepts	4
Retrieve_documents_MR1	1	Get CVs	4
EBI_ClustalW2	2	Get Evidence Types	4
EBI_ClustalW2_phylogentic_tree	2	Get graphs	4
EBI_Kalign	3	Get Graphs of Name	4
EBI_TCoffee	3	Get relations	4

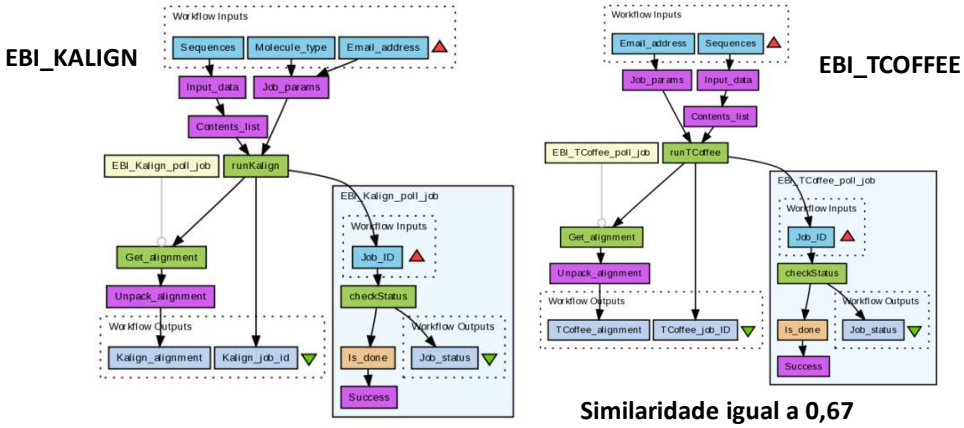

Figura 3. *Workflows* similares analisados pela arquitetura SimiFlow

Como análise desse estudo, pode-se apontar a necessidade de usar a distribuição e o paralelismo para a realização dos agrupamentos, uma vez que o cartucho da comparação é pesado computacionalmente, devido à existência de um número grande de comparações, dependendo do número de *workflows* na base. Além disso, apesar de o escopo desse segundo estudo não ter sido a identificação dos agrupamentos, verificou-se que alguns *workflows* foram agrupados erroneamente. Logo, o algoritmo ainda possui alguns pontos que precisam ser refinados.

4. Trabalhos Relacionados

A proposta de Santos *et al.* (2008) é a que mais se assemelha à arquitetura SimiFlow no que diz respeito ao objetivo de realizar o agrupamento de *workflows*. Nesta abordagem, os *workflows* são representados tanto como grafos, quanto como vetores multidimensionais. Além disso, para o cálculo da similaridade, utiliza-se a medida do máximo subgrafo comum (Bunke e Shearer 1998) para as representações em grafos, e a distância do cosseno para as representações por vetor. Em relação ao algoritmo de agrupamento, foram realizados testes com os algoritmos K-médias, K-medóides e algoritmo hierárquico (Jain et al. 1999). Apesar dos resultados serem satisfatórios, as representações por grafo e por vetor não conseguem obter as informações necessárias para realizar a comparação entre os *workflows*. O SimiFlow, por outro lado, verifica cada componente de um *workflow*, como portas de entrada, portas de saída, parâmetros, entre outros, o que deixa o cálculo da similaridade mais robusto. É importante ressaltar que as linhas de experimentos trabalham tanto com o nível abstrato, como com o nível concreto, o que também constitui uma vantagem.

Existem alguns trabalhos que apresentam técnicas para a comparação de modelos de dados. Ohst *et al.* (2003) mostra um método de computar as diferenças entre versões de um documento UML. Nesse método, um diagrama UML é representado por um grafo direcionado acíclico, o qual forma uma árvore de abrangência (*spanning tree*). Para que a comparação entre as versões seja feita, os níveis das árvores são comparados entre si. A semelhança entre esse método e o SimiFlow diz respeito a esse nivelamento. No SimiFlow, cada atividade é comparada com todas as demais, e cada porta é comparada com todas as demais, ou seja, a comparação é feita por níveis. A comparação por níveis permite que o cálculo da similaridade seja mais robusto e preciso. A diferença entre os métodos está na estruturação do modelo de dados: os *workflows* no cartucho não estão representados por grafos. Pode-se citar ainda o trabalho de Uhrig (2008), cujo objetivo é igualar dois diagramas de classes que sejam diferentes entre si usando o menor custo possível. Para isso, cada classe de um diagrama é analisada contra todas as classes do outro diagrama, e um grafo bipartido é construído, onde os pesos das arestas seriam os custos associados a cada par. Formulando como um problema de otimização, o algoritmo visa o custo mínimo associado ao grafo.

O método de comparação que mais se assemelha ao apresentado neste trabalho é aquele usado pelo SiDiff (2010). O objetivo do SiDiff é comparar modelos de dados, proporcionando mecanismos de diferenciação e combinação (*merge*), por exemplo. O algoritmo de comparação utilizado pelo SiDiff, assim como em Ohst *et al.* (2003), usa árvores para realizar a comparação por níveis. Porém, a diferença entre eles, e a semelhança entre o SiDiff e o SimiFlow, é a existência de pesos que ponderam a semelhança entre os elementos, o que é, de certa forma, equivalente à penalidade apresentada na Seção 3.3, apesar da forma de cálculo ser diferente. O SimiFlow, contudo, agrega a semântica de *workflows* científicos, levando em consideração detalhes dos elementos que são mais significativos para o trabalho de agrupamento.

5. Conclusão

A comparação por similaridade e o agrupamento de *workflows* pode facilitar a organização de bases extensas de *workflows* científicos. Tal organização pode ser realizada *a posteriori* a partir dos agrupamentos encontrados, tendo como fator de agrupamento a similaridade entre os *workflows*. A arquitetura SimiFlow fornece uma infraestrutura para estes recursos.

Através do primeiro estudo realizado, verificou-se que a arquitetura consegue identificar agrupamentos de *workflows* que são, de fato, semelhantes. Além disso, a arquitetura também possui uma boa escalabilidade, pois consegue realizar as comparações e os agrupamentos com um número grande de *workflows*, como mostrado pelo segundo estudo. Entretanto, a arquitetura SimiFlow ainda apresenta pontos a serem melhorados. Foi observado, a partir do segundo estudo, que a calibragem das penalizações no processo de comparação por similaridade precisa ser refinada. O agrupamento em si também precisa de ajustes, que já estão em andamento.

Como trabalho futuro, pode-se incluir o teste com outros algoritmos de comparação e de agrupamento, e o uso do paralelismo para melhorar o desempenho do processo de comparação. A partir do agrupamento, espera-se identificar conjuntos de *workflows* que são candidatos a serem representados por linhas de experimentos, que criam a base para a futura construção de linhas de experimentos a partir de *workflows* previamente estabelecidos, ou seja, através da abordagem ascendente.

Agradecimentos. Os autores gostariam de agradecer ao CNPq pelo financiamento parcial deste trabalho. Vítor Silva foi financiado por uma bolsa PIBIC/CNPq.

Referências

Altintas, I., Berkley, C., Jaeger, E., Jones, M., Ludascher, B., Mock, S., (2004), "Kepler: an extensible system for design and execution of scientific workflows". In: *SSDBM*, p. 423-424, Greece.

Bunke, H., Shearer, K., (1998), "A graph distance metric based on the maximal common subgraph", *Pattern Recogn. Lett.*, v. 19, n. 3-4, p. 255-259.

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 2006*, p. 745-747, USA.

Cavalcanti, M. C., Targino, R., Baião, F., Rössle, S. C., Bisch, P. M., Pires, P. F., Campos, M. L. M., Mattoso, M., (2005), "Managing structural genomic workflows using web services", *Data & Knowledge Engineering*, v. 53, n. 1, p. 45-74.

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.

GExp, (2009), *Brazilian project for supporting large scale management of scientific experiments*, http://gexp.nacad.ufrj.br/.

Goble, C. A., Roure, D. C. D., (2007), "myExperiment: social networking for workflow-using e-scientists". In: *Proceedings of the 2nd workshop on Workflows in support of large-scale science*, p. 1-2, Monterey, California, USA.

Jain, A. K., Murty, M. N., Flynn, P. J., (1999), "Data clustering: a review", *ACM Comput. Surv.*, v. 31, n. 3, p. 264-323.

Larman, C., (2004), *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development*. 3 ed. Prentice Hall PTR.

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

Ogasawara, E., Paulino, C., Murta, L., Werner, C., Mattoso, M., (2009), "Experiment Line: Software Reuse in Scientific Workflows". In: *21th SSDBM*, p. 264–272, New Orleans, LA.

Ohst, D., Welle, M., Kelter, U., (2003), "Differences between versions of UML diagrams". In: *Proc. 9th ESEC*, p. 227-236, Helsinki, Finland.

Oinn, T., Addis, M., Ferris, J., Marvin, D., Senger, M., Greenwood, M., Carver, T., Glover, K., Pocock, M. R., et al., (2004), "Taverna: a tool for the composition and enactment of bioinformatics workflows", *Bioinformatics*, v. 20, p. 3045-3054.

Santos, E., Lins, L., Ahrens, J. P., Freire, J., Silva, C. T., (2008), "A First Study on Clustering Collections of Workflow Graphs", *Proc. IPAW 2008*, Springer-Verlag, p. 160-173.

Seo, J., Seno, S., Takenaka, Y., Matsuda, H., (2007), "Retrieving Functionally Similar Bioinformatics Workflows Using TF-IDF Filtering", *IPSJ Digital Courier*, v. 3, n. 0, p. 164–173.

SiDiff, (2010), *SiDiff*, http://www.sidiff.org.

Uhrig, S., (2008), "Matching class diagrams: with estimated costs towards the exact solution?". In: *Proc. 2008 CVSM*, p. 7-12, Leipzig.