Aplicação de Árvores de Decisão para Recomendação de Parâmetros em *Workflows* Científicos

Renan Vinagre Câmara, Aline Paes, Daniel de Oliveira

Instituto de Computação – Universidade Federal Fluminense (UFF)
Niterói – RJ – Brasil

{alinepaes,danielcmo}@ic.uff.br

Resumo. *Diversos experimentos de larga escala modelados como workflows científicos podem executar em paralelo por diversos dias ou semanas em ambientes de alto desempenho. O tempo de execução é determinado por fatores como o volume de dados de entrada, a quantidade de parâmetros explorados, etc. Assim, se torna importante para o cientista que as execuções de workflows que não produzem resultados satisfatórios ou que produzem resultados com erros sejam reduzidas ao máximo. Estimar quais execuções irão falhar (ou não) é um problema importante, porém em aberto. De forma a reduzir esse problema, propomos um mecanismo de recomendação de parâmetros para workflows baseado em algoritmos de mineração de dados para que o cientista possa configurar seu workflow da melhor forma possível (e.g., para evitar erros) antes da execução propriamente dita.*

1. Introdução

Na última década, os *workflows* científicos tornaram-se um padrão *de fato* para modelagem de experimentos científicos que são intensivos tanto em termos computacionais quanto em termos de volume de dados produzidos (Mattoso *et al*., 2010). Os w*orkflows* científicos são abstrações que representam o fluxo de dados entre várias atividades (*i.e*., programas ou invocações de serviços). Os Sistemas de Gerência de *Workflows* Científicos (SGWfCs) são responsáveis por modelar, executar e monitorar tais *workflows*. Além disso, esses sistemas coletam dados históricos sobre os experimentos, chamados de dados de proveniência (Freire *et al*., 2008). Exemplos de SGWfC são: o Kepler (Altintas *et al*., 2004), o Swift/T (Wozniak *et al*., 2013) e o SciCumulus (Oliveira *et al*., 2010). Estes SGWfC são aplicados com sucesso em várias áreas, tais como a biologia (Ocaña *et al*., 2011) e astronomia (Jacob *et al*., 2009).

Diante do crescente aumento da complexidade dos *workflows*, seja em termos de quantidade de atividades ou da quantidade de dados de entrada a serem processados, muitos *workflows* científicos são executados em ambientes de processamento de alto desempenho (PAD) como *clusters*, grades e nuvens de computadores, a fim de reduzir o tempo total de processamento. Entretanto, mesmo nesse tipo de ambiente, uma execução completa do *workflow* pode se estender por vários dias, dependendo do volume de dados de entrada, quantidade de parâmetros explorados, quantidade de atividades, *etc*. Esse cenário torna-se ainda mais crítico se considerarmos que em determinados ambientes o usuário paga pelo serviço, então a demora na execução implica em custos financeiros (muitas vezes elevados).

Assim, é muito importante para o cientista que a quantidade de execuções dos *workflows* e suas respectivas atividades que não produzem resultados relevantes seja reduzida ao máximo. Técnicas de condução do *workflow* (*i.e., steering*) já vem sendo aplicadas de forma que o cientista possa analisar os resultados intermediários em tempo de execução e executar ações a partir de tais resultados, como mudar um valor de parâmetro ou até mesmo parar a execução do *workflow*. Entretanto, mesmo com o uso de técnicas de condução do *workflow*, o cientista ainda deve esperar que resultados intermediários sejam produzidos, o que também pode levar um tempo considerável. Tomemos como exemplo *workflow* de análise filogenética SciPhy (Ocaña *et al*., 2011). Das 4 atividades do SciPhy (Alinhamento Múltiplo de Sequência, Conversão, Eleição do Modelo Evolutivo e Geração da Árvore), as duas últimas são as mais custosas em termos computacionais, usualmente levando horas para serem finalizadas. Ou seja, para que o cientista possa analisar os resultados em tempo real nesse caso, ele necessitará aguardar algumas horas e caso os resultados não sejam satisfatórios, esse tempo de execução (e seu possível custo financeiro associado) terá sido desperdiçado.

Dessa forma, se torna fundamental que o cientista possa configurar a execução do *workflow* (*e.g.,* definir parâmetros, dados de entrada, *etc*.) da melhor forma possível antes da sua execução. Atualmente, esse processo de configuração do *workflow* ocorre de forma *ad-hoc*. Para reduzir esse problema, neste trabalho apresentamos um estudo inicial de viabilidade para desenvolver um mecanismo de recomendação para auxiliar a configuração do *workflow*. Existem diversos mecanismos de recomendação propostos na literatura (Ellkvist *et al*., 2008; Koop *et al*., 2008; Oliveira *et al*., 2008), porém os mesmos tem como objetivo sugerir uma sequência de programas coerente a ser utilizada baseada em *workflows* previamente modelados, sem se preocupar com o desempenho do *workflow* sugerido. Apesar de importante, esse não é o único tipo de recomendação útil para os cientistas.

No caso de *workflows* que executam em paralelo em ambientes de PAD, os valores dos parâmetros e os dados de entrada fornecidos para uma determinada execução do *workflow* podem gerar tanto impacto na qualidade da saída e no tempo de execução quanto a escolha da sequência de atividades. Dessa forma, os parâmetros e os dados de entrada devem ser escolhidos da melhor maneira possível de forma que menos erros ocorram e que o tempo de execução seja reduzido. Assim, neste artigo apresentamos um primeiro estudo sobre a recomendação de parâmetros e dados de entrada para *workflows*. Para tanto, são utilizados dados de proveniência enriquecidos com dados de domínio e algoritmos de classificação e regressão (Han *et al*., 2011) já bastante difundidos na área de mineração de dados. A ideia principal é extrair dados relativos a tempos de execução, valores de parâmetros e dados de domínio do banco de dados de proveniência para que possamos criar modelos e regras que auxiliem o cientista na configuração da execução do *workflow*. Nos experimentos realizados utilizamos *workflows* executados no SGWfC SciCumulus e sua base de proveniência que segue o modelo PROV-Wf (Costa *et al*., 2013; Oliveira *et al*., 2014). A motivação deste artigo, portanto, está na possibilidade de reduzir o tempo e o custo financeiro de execuções de *workflows* por meio do histórico de *workflows* previamente executados, de forma que o sucesso de execuções passadas ajude a melhorar o desempenho dos *workflows* que serão executados no futuro.

Este texto está organizado da seguinte forma: A Seção 2 discute os trabalhos relacionados. A Seção 3 descreve a abordagem algébrica para representação de

workflows que é ponto chave do trabalho proposto nesse artigo. A Seção 4 apresenta as análises realizadas para recomendação, e, finalmente, a Seção 5 conclui o artigo.

2. Trabalhos Relacionados

Conforme discutido na introdução deste artigo, existem diversos trabalhos associados ao tema de recomendação de *workflows* na literatura. Entretanto, diferente da proposta do presente artigo, esses trabalhos tem como foco a sugestão do encadeamento das atividades do *workflow* e não a configuração de parâmetros ou dados de entrada.

McDowall e Kerschberg (2012) propõem uma técnica para composição automática de serviços baseada em uma ontologia que descreve os serviços disponíveis, seus parâmetros de entrada e de saída. A recomendação da atividade é realizada por meio de inferências na ontologia proposta. Yeo e Abidi (2013) propõem a recomendação de um conjunto de atividades por meio do cálculo de similaridade estrutural de dois (ou mais) *workflows*. Os autores utilizam como métricas os tipos de atividades, suas conexões e portas, mas sem especificar valores de parâmetros.

Os artigos de Zeng *et al*. (2011), Oliveira *et al*. (2008) e Koop *et al*. (2008) propõem abordagens que sugerem as combinações de atividades mais frequentes, *i.e*. combinações que são utilizadas pela maioria dos cientistas, de forma a otimizar o processo de modelagem de *workflows* e melhorar a qualidade dos *workflows* desenvolvidos. O que difere as abordagens é a técnica utilizada. Enquanto Oliveira *et al*. e Zeng *et al*. fazem uso de regras de associação para que se possa predizer quais programas devem ser combinados, a abordagem de Koop *et al*. não se baseia em nenhuma técnica de mineração de dados (utiliza apenas a frequência de aparecimento da sequência), o que faz com que a abordagem proposta falhe na recomendação de *workflows* que contêm atividades que foram pouco utilizadas anteriormente.

3. Abordagem Algébrica para Representação de *Workflows*

A base para os estudos de recomendação de parâmetros em *workflows* científicos apresentado nesse artigo é a modelagem do *workflow* utilizando uma álgebra de *workflows* (Ogasawara *et al*., 2011). Essa álgebra rege o SGWfC SciCumulus, e usa relações para armazenar a especificação do *workflow* (proveniência prospectiva) e dados de execução (proveniência retrospectiva), além dos dados específicos de domínio que são extraídos de arquivos produzidos. A abordagem algébrica é baseada em um conjunto de operadores alicerçados na álgebra relacional de banco de dados (*Map*, *Reduce*, *Filter*, *SplitMap*, *MRQuery* e *SRQuery*), que consideram tanto as atividades do *workflow* e os dados como operandos. Cada atividade está associada a um operador, que tem como operandos as relações de entrada. Todos os resultados da execução são armazenados em uma relação de saída, que servirá como a relação de entrada para o operador associado à atividade seguinte. No contexto deste artigo, a maior vantagem apresentada por tal abordagem é a representação dos dados e parâmetros por meio de relações, o que torna simples sua extração e consulta.

De forma a apresentar o potencial da abordagem algébrica para extração dos dados que são utilizados como entrada para os algoritmos de mineração, vamos mostrar a visão algébrica do *workflow* SciPhy (2011), para análises filogenéticas. O *workflow* SciPhy é composto por 4 atividades implementadas pelos seguintes programas MAFFT (Alinhamento Múltiplo de Sequências), Readseq (Conversão), ModelGenerator (Escolha do Modelo Evolutivo) e RAxML (Geração da Árvore Filogenética). Mais informações do SciPhy podem ser obtidas em Ocaña *et al*. (2011). Uma execução

completa do SciPhy pode durar mais de 24 horas dependendo dos dados de entrada. Para explicar a abordagem algébrica e o *workflow* SciPhy de forma didática, optamos por simplificá-lo aqui e mostrar as relações algébricas apenas para as atividades mais representativas do *workflow*: ModelGenerator e RAxML.

O fragmento relativo as atividades ModelGenerator e RAxML do SciPhy se encontra representado na Figura 1. Nesse fragmento temos 3 relações envolvidas: R_{iMG}, R_{oMG} e R_{oRAxML}. Os parâmetros de cada atividade do *workflow* são representados como os atributos das relações. Por exemplo, para cada uma das *n* tuplas da relação de entrada R_{iMG} o programa ModelGenerator é executado com diferentes valores para os três parâmetros (*file*, *num* e *length*, sendo os dois últimos previamente extraídos de arquivos produzidos) gerando *n* tuplas na relação de saída R_{oMG}. Cada tupla de R_{oMG} está relacionada com uma tupla de R_{iMG} com campos específicos de domínio que foram extraídos de arquivos produzidos, como por exemplo *model* e *bootstrap*. Para cada tupla de R_{oMG}, o programa RAxML é executado e o resultado é armazenado na relação saída R_{oRAxML}.

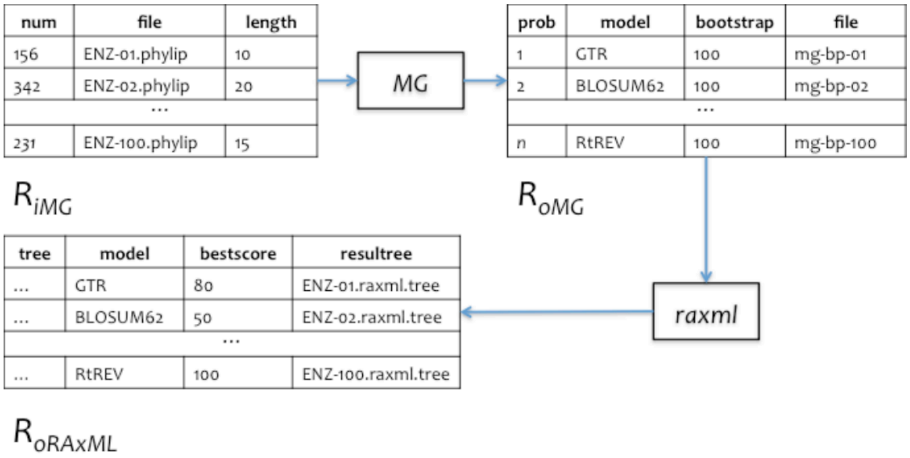

Figura 1 Representação algébrica do *Workflow* SciPhy

Utilizando a modelagem algébrica, os cientistas estão aptos a consultar qualquer uma das relações em tempo de execução, enquanto as tuplas de saída ainda estão sendo geradas. Por exemplo, uma consulta interessante pode ter como objetivo recuperar todas as execuções do *workflow* e quais os valores dos parâmetros *num* (número de alinhamentos), *length* (tamanho da maior sequência alinhada) e *model* (modelo evolutivo escolhido), que foram gerados a partir de um valor de atributo *file* (nome do arquivo que foi processado) em conjunto com o tempo de execução da atividade. Um outro exemplo poderia ser retornar quais os modelos evolutivos escolhidos que geram árvores filogenéticas com *bestscore* ≥ 90. Apesar de úteis, essas consultas não auxiliam os cientistas na configuração dos parâmetros, elas apenas possibilitam o monitoramento em tempo real da execução ou uma análise *post mortem*. Dessa forma, apenas as consultas ao banco de dados de proveniência não são suficientes. É desejável que existam modelos bem definidos que possam atuar como guias aos cientistas para configurar a execução do seu *workflow a priori*. Para tanto, os dados extraídos das consultas devem ser utilizados como insumo para algoritmos de classificação e regressão para que possamos desenvolver modelos capazes de orientar cientistas na configuração de seus *workflows*.

4. Recomendação de Parâmetros em *Workflows* Científicos

O objetivo deste artigo é mostrar a viabilidade para o desenvolvimento de uma abordagem para a recomendação de parâmetros em *workflows* científicos por meio da criação de uma série de modelos que possam auxiliar cientistas na configuração de seus *workflows*. Dessa forma, utilizamos o *workflow* SciPhy executado no SGWfC SciCumulus como estudo de caso. A modelagem do SciPhy é a mesma apresentada em Ocaña *et al*. (2011) e no fragmento de *workflow* apresentado na Figura 1. Após a execução do SciPhy, o banco de dados de proveniência do SciCumulus foi consultado para extração dos dados que servirão de insumo para os algoritmos de classificação e regressão. Devido a restrições de espaço no artigo, elegemos dois tipos de recomendação de parâmetros para *workflows* executados em ambientes de PAD: (i) Qual a previsão de erro para execuções com os valores de parâmetros informados? e (ii) Qual a previsão de tempo de execução para os valores de parâmetros informados? Ambas as consultas são fundamentais para que o cientista poupe tempo e recursos financeiros, já que evitam execuções desnecessárias para alguns conjuntos de dados e parâmetros.

Como as relações que contêm os valores de parâmetros (consumidos e produzidos) e os dados de domínio (*e.g.*, R_{iMG}, R_{oMG} e R_{oRAxML}) se encontram no mesmo banco de dados de proveniência, as consultas de extração podem ser representadas como simples comandos SQL. As relações seguem o modelo PROV-Wf (Costa *et al*., 2013), uma extensão do padrão W3C PROV (Groth e Moreau, 2013). Usando um banco de proveniência que segue o PROV-Wf, os cientistas são capazes de consultar dados de execução como o tempo de execução de cada atividade, dados de domínio, *etc*. O PROV-Wf é composto por um conjunto de classes (cada classe é mapeada para uma tabela dentro do banco de dados de proveniência) para representar conceitos específicos do *workflow*. No processo de extração dos dados utilizamos algumas das tabelas (além das já apresentadas na Figura 1): *hworkflow* que contém metadados do *workflow*, *hactivity* que contém metadados de cada atividade e *hactivation* que contém metadados das execuções de atividades. As tabelas *iModelGenerator*, *oModelGenerator* e *oRAxMl* contêm os campos descritos nas relações R_{iMG}, R_{oMG} e R_{oRAxML} (respectivamente) mais uma chave estrangeira para a tabela *hactivation* que associa a cada combinação de parâmetros a execução que a produziu/consumiu. Todos os dados foram extraídos de experimentos executados no ambiente de nuvem Amazon EC2 usando 16 núcleos (8 máquinas virtuais do tipo *m1.large*). Todos os dados de proveniência foram armazenados no SGBD PostgreSQL. A execução do SciPhy para 250 arquivos de entrada (arquivos multi-fasta contendo sequências de DNA e RNA) gerou 1.000 execuções de atividades. Isso significa que, para cada execução de SciPhy 1 registro é inserido na tabela *hworkflow*, 4 em *hactivity*, 1.000 em *hactivation*, e 250 em cada tabela de relação (*iModelGenerator*, *oModelGenerator* e *oRAxMl*). A consulta a seguir foi implementada de forma a extrair os dados necessários para que se possa atender os tipos de recomendação (i) e (ii) apresentadas anteriormente:

```
SELECT extract (epoch from (t1.endtime-t2.starttime)) as
time, omg.num, omg.length, omg.model FROM hworkflow w,
hactivity a, hactivation t1, hactivation t2 oModelGenerator
omg, oRAxML ora WHERE w.wkfid = a.wkfid AND a.actid =
t1.actid AND t1.taskid = omg.nexttaskid AND t1.taskid =
ora.taskid AND omg.previoustaskid = t2.taskid AND w.tag
like '%SciPhy%'
```

A consulta retornou 250 tuplas. Para cada uma das execuções do fragmento do *workflow* composto pelos programas ModelGenerator e RAxML possuímos o tempo de execução e os valores dos parâmetros *num*, *length* e *model* associados a cada execução. A partir desses dados, utilizamos algoritmos de mineração de dados implementados no *toolkit* Orange (http://orange.biolab.si/) para criar modelos capazes de prever o comportamento do *workflow* para um determinado conjunto de valores de parâmetros e recomendar (ou não) a execução do mesmo. O algoritmo de mineração demonstrado aqui infere Árvores de Decisão, tanto para atender a uma tarefa de classificação, no caso de valores de parâmetros discretos (*e.g.,* se houve erro ou não), como de regressão, quando os parâmetros são contínuos (*e.g.,* tempo de execução) (Han *et al*., 2011). Árvores de decisão tomam como entrada uma situação descrita por um conjunto de atributos (no nosso caso os valores extraídos do banco de dados de proveniência) e retornam uma decisão, que é um valor previsto de saída de acordo com o valor de entrada. Uma árvore de decisão pode ser mapeada diretamente para um conjunto de regras SE-ENTÃO, o que facilita o seu entendimento por parte dos cientistas.

No experimento conduzido neste artigo, criamos dois modelos de recomendação baseado em árvores de decisão. No primeiro, relativo ao tipo de recomendação (i), definimos quais combinações de parâmetros *num*, *length* e *model* têm maior (ou menor) probabilidade de que a execução falhe. Como o cientista tem que informar *a priori* quais combinações de parâmetros gostaria de explorar (via interface do SGWfC), podemos dessa forma indicar em quais delas há a chance de a execução falhar. A árvore gerada pelo algoritmo é apresentada na Figura 2, onde podemos verificar que os valores do parâmetro *model* quando iguais a DCMut+G+F e RtREV+G+F historicamente apresentam falhas na execução do *workflow*. Essas falhas não estão somente associadas ao modelo escolhido, mas também ao número de alinhamentos ($num > 21$) e ao tamanho ($length \leq 1.413{,}5$). Dessa forma, se o cientista informasse uma combinação de valores de parâmetros do tipo (*model*, *num*, *length*) = (RtREV+G+F, 45, 2.500) o modelo indicaria que há chance de a execução falhar.

No segundo experimento, relativo ao tipo de recomendação (ii), definimos quais combinações de parâmetros *num*, *length* e *model* têm maior (ou menor) tempo de execução associado. Assim como no experimento anterior, o cientista tem que informar *a priori* quais combinações de parâmetros gostaria de explorar, podemos dessa forma ter uma expectativa do tempo de processamento de cada combinação dessas. A árvore gerada é apresentada na Figura 3, onde podemos verificar que os valores do parâmetro *num* quando superiores a 185 historicamente apresentam um tempo de execução médio de 385 minutos. A estimativa do tempo de execução não está somente associada ao número de alinhamentos, mas também ao tamanho das sequências que foram fornecidas de entrada. Por exemplo, se o cientista informasse uma combinação de valores de parâmetros do tipo (*num*, *length*) = (100,15) o modelo indicaria que o tempo médio de processamento dessa combinação de valores seria de 44 minutos.

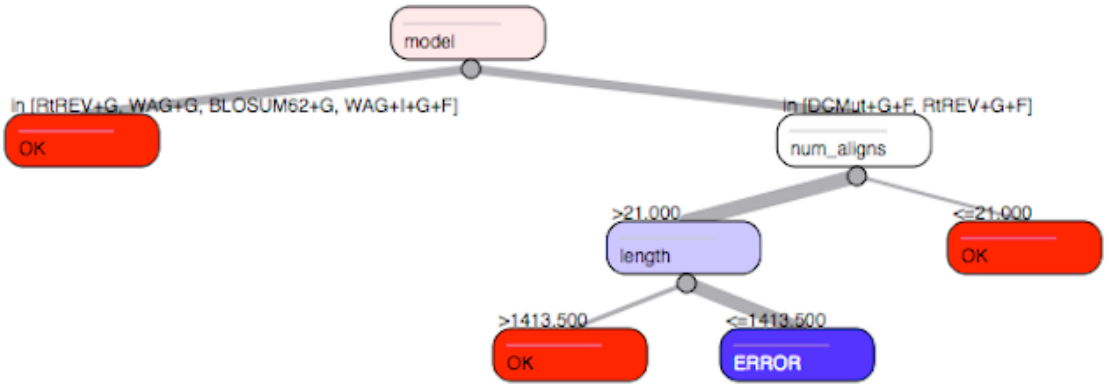

Figura 2 Árvore de decisão relativa falhas na execução do *workflow* científico

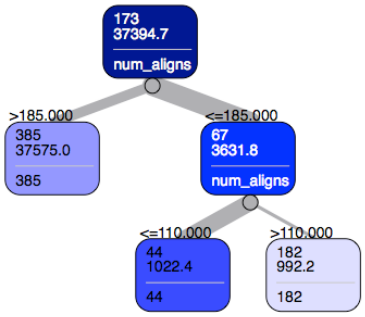

Figura 3 Árvore de decisão relativa ao tempo de execução do *workflow*

Apesar de iniciais, os resultados obtidos indicam que é possível prever o tempo médio de execução do *workflow* e a possibilidade de falhas a partir de uma determinada combinação de valores de parâmetros uma vez que tenhamos a disposição modelos preditivos como os apresentados neste artigo.

5. Conclusão

Até que se alcance o resultado desejado de um experimento, é comum que os cientistas avaliem uma grande gama de combinações de valores de parâmetros. Dependendo do ambiente de execução e dos aspectos mais valorizados pelo cientista no momento da execução, uma combinação pode ser mais vantajosa do que outra em termos de desempenho ou taxa de falhas. Executar todas as possíveis combinações pode ser computacionalmente e financeiramente custoso. Diante deste cenário, é importante que o cientista tenha uma estimativa *a priori* se uma execução tem chance de falhar e quanto tempo em média levará. Nesse artigo apresentamos um primeiro experimento sobre recomendação de parâmetros em *workflows* científicos. Tal abordagem permite que os cientistas avaliem as combinações de parâmetros fornecidas de modo a tentar diminuir a incidência de erros e reduzir o tempo de execução. A abordagem proposta extrai dados de proveniência e específicos de domínio de um banco de dados unificado de proveniência e por meio de algoritmos de mineração de dados gera modelos capazes de fornecer tais previsões para o cientista. Os resultados indicam que é possível prever o tempo de execução do *workflow* e a chance de falhas a partir de uma determinada combinação de valores de parâmetros. Trabalhos futuros incluem a avaliação de novas técnicas de predição e a validação dos modelos gerados.

Agradecimentos. Os autores gostariam de agradecer ao CNPq e a FAPERJ pelo financiamento parcial deste trabalho.

Referências Bibliográficas

Altintas, I., Berkley, C., Jaeger, E., Jones, M., 2004. Kepler: An Extensible System for Design and Execution of Scientific *Workflows*.

Costa, F., Silva, V., de Oliveira, D., Ocaña, K., Ogasawara, E., Dias, J., Mattoso, M., 2013. Capturing and Querying *Workflow* Runtime Provenance with PROV: A Practical Approach, in: Proceedings of the EDBT 2013, New York, NY, USA, pp. 282–289.

Ellkvist, T., Koop, D., Anderson, E.W., Freire, J., Silva, C., 2008. Using Provenance to Support Real-Time Collaborative Design of *Workflows*, in: IPAW 2008, Salt Lake City, UT, USA, LNCS. Springer-Verlag, pp. 266–279.

Freire, J., Koop, D., Santos, E., Silva, C.T., 2008. Provenance for Computational Tasks: A Survey. Computing in Science Engineering 10, 11–21.

Groth, P., Moreau, L., 2013. W3C PROV - An Overview of the PROV Family of Documents [WWW Document]. URL http://www.w3.org/TR/2013/ NOTE-prov-overview-20130430

Han, J., Kamber, M., Pei, J., 2011. Data Mining: Concepts and Techniques, Third Edition, 3rd ed. Morgan Kaufmann.

Jacob, J.C., Katz, D.S., Berriman, G.B., Good, J.C., Laity, A.C., Deelman, E., Kesselman, C., Singh, G., Su, M.-H.,Prince, T.A., Williams, R., 2009. Montage: a grid portal and software toolkit for science-grade astronomical image mosaicking. International Journal of Computational Science and Engineering (IJCSE) 4, 73–87.

Koop, D., Scheidegger, C.E., Callahan, S.P., Freire, J., Silva, C.T., 2008. VisComplete: Automating Suggestions for Visualization Pipelines. IEEE Transactions on Visualization and Computer Graphics 14, 1691–1698.

Mattoso, M., Werner, C., Travassos, G.H., Braganholo, V., Ogasawara, E., Oliveira, D.D., Cruz, S.M.S.D., Martinho, W., Murta, L., 2010. Towards supporting the life cycle of large scale scientific experiments. International Journal of Business Process Integration and Management 5, 79.

McDowall, J., Kerschberg, L., 2012. Agent Negotiation Strategies for Composing Service *Workflows*, in: 2012 IEEE 28th International Conference on Data Engineering Workshops (ICDEW).

Ocaña, K.A.C.S., Oliveira, D. de, Ogasawara, E., Dávila, A.M.R., Lima, A.A.B., Mattoso, M., 2011. SciPhy: A CloudBased *Workflow* for Phylogenetic Analysis of Drug Targets in Protozoan Genomes, Simpósio Brasileiro de Bioinformática, 2011

Telles, G.P., Palakal, M. (Eds.), Advances in Bioinformatics and Computational Biology, Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 66–70.

Ogasawara, E., Dias, J., Oliveira, D., Porto, F., Valduriez, P., Mattoso, M., 2011. An Algebraic Approach for DataCentric Scientific *Workflows*. Proceedings of the 37th International Conference on Very Large Data Bases (PVLDB) 4, 1328–1339.

Oliveira, D., Ogasawara, E., Baião, F., Mattoso, M., 2010. SciCumulus: A Lightweight Cloud Middleware to Explore Many Task Computing Paradigm in Scientific *Workflows*, in: Proceedings of the 3rd International Conference on Cloud Computing, CLOUD '10. IEEE Computer Society, Washington, DC, USA.

Oliveira, F., Murta, L., Werner, C., Mattoso, M., 2008. Using Provenance to Improve *Workflow* Design, in: IPAW. Salt Lake City, UT, USA, pp. 136 – 143.

Oliveira, W., Oliveira, D., Braganholo, V., 2014. Experiencing PROV-Wf for Provenance Interoperability in SWfMSs, in: IPAW. Cologne, German.

Wozniak, J.M., Armstrong, T.G., Wilde, M., Katz, D.S., Lusk, E., Foster, I.T., 2013. Swift/T: Large-Scale Application Composition via Distributed-Memory Dataflow Processing, in: Proceedings of the 13th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid). IEEE, pp. 95–102.

Yeo, P., Abidi, S.S.R., 2013. Dataflow Oriented Similarity Matching for Scientific *Workflows*, in: Parallel and Distributed Processing Symposium Workshops PhD Forum (IPDPSW), 2013 IEEE 27th International.

Zeng, R., He, X., Aalst, W.M.P. van der, 2011. A Method to Mine *Workflows* from Provenance for Assisting Scientific *Workflow* Composition, in: 2014 IEEE World Congress on Services. IEEE Computer Society, Los Alamitos, CA, USA.