$OntoExpLine$: Rumo a uma Ontologia para Representação de Linhas de Experimento Algébricas*

Luiz Gustavo Dias[1], Bruno Lopes[1], Daniel de Oliveira[1],

[1] Instituto de Computação – Universidade Federal Fluminense (UFF)

lgdias@id.uff.br, {bruno,danielcmo}@ic.uff.br

Abstract. *Scientific workflows are used as an abstraction to implement complex simulations of Computational Science and Engineering (CSE). However, managing CSE experiments is not a simple task because the same experiment can involve the execution and correlated analysis of several workflows. The algebraic experiment lines (LinExps) involve techniques to model CSE experiments at different levels of abstraction and, despite representing a step forward, they still lack support about domain data and automatic checking that assist workflow derivation process. This paper proposes the OntoExpLine, a LinExps ontology that aims at providing semantics and flexibility to the scientific experimentation process.*

Resumo. *Workflows científicos são usados como uma abstração para implementar simulações complexas de Ciência Computacional e Engenharia (CSE). Entretanto, gerenciar experimentos de CSE não é uma tarefa simples pois um mesmo experimento pode envolver a execução e a análise correlacionada de vários workflows. As linhas de experimentos algébricas (LinExps) envolvem técnicas para modelar experimentos CSE em diferentes níveis de abstração e apesar de representar um avanço, ainda carecem de apoio em relação a dados de domínio e checagens automáticas que auxiliem processos de derivaçãode workflows. Este artigo propõe a $OntoExpLine$, uma ontologia de LinExps que busca oferecer maior semântica e flexibilidade ao processo de experimentação científica.*

1. Introdução

Os experimentos de Ciência Computacional e Engenharia (CSE) são baseados em modelos computacionais que resolvem problemas que normalmente requerem Processamento de Alto Desempenho (PAD) [Sameh et al. 1996]. Em diversos casos, esses modelos são implementados na forma de *workflows* científicos (chamados apenas de *workflows*), e podem ser definidos como uma abstração que representa as etapas de um experimento de CSE. Seu uso se torna vantajoso pois encapsula a complexidade envolvida no desenvolvimento de experimentos de CSE, uma vez que exigem que processos que antecedem sua execução sejam bem planejados, visando tanto a qualidade dos produtos de dados, quanto viabilidade de custos. *Workflows* são executados por sistemas complexos chamados de Sistemas de Gerência de *Workflows* (SGWf).

Entretanto, um experimento de CSE envolve não somente uma execução de um *workflow*. Um experimento pode considerar não só a exploração de várias combinações de dados e parâmetros de entrada, mas também a alteração de atividades do *workflow* para explorar diferentes métodos e ferramentas de forma a confirmar ou refutar uma determinada hipótese. Apesar dos SGWfs existentes oferecerem apoio para execução de *workflows*, os mesmos ainda carecem de um apoio no nível do experimento. Em um SGWf não é possível associar as diversas

*O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Código de Financiamento 001. O trabalho foi financiado com recursos do CNPq e FAPERJ.

variações de um *workflow* com um mesmo experimento científico. Esse tipo de conhecimento ainda é tácito, uma vez que apenas as pessoas envolvidas no processo científico estão cientes do mesmo. Essa falta de apoio no nível do experimento pode levar o cientista a reexecutar *workflows* sem necessidade ou a não avaliar uma determinada combinação de parâmetros importante. O problema de representar experimentos científicos em vários níveis de abstração é abordado em diversos trabalhos na literatura [Gil et al. 2011, de Oliveira et al. 2012, Carvalho et al. 2018], pois sem uma representação em níveis mais altos de abstração os cientistas não conseguem correlacionar resultados de múltiplas execuções de *workflows*. Um grande problema nessa representação é como associar dados de proveniência prospectiva e retrospectiva [Freire et al. 2008]. Uma das abordagens propostas para representar *workflows* em níveis mais altos de abstração são as Linhas de Experimento (LinExp) Algébricas [Marinho et al. 2017].

O conceito de LinExp objetiva auxiliar o cientista no processo de derivação de *workflows* executáveis a partir de uma definição abstrata. Uma LinExp modela todas as possíveis variações de programas que implementam uma atividade, quais atividades são opcionais, seus parâmetros, dependências e restrições encontradas em um determinado experimento de CSE. A partir dessa representação, *workflows* executáveis podem ser derivados pelos cientistas [Marinho et al. 2017]. Apesar das LinExps representarem um avanço, seu processo de criação não é trivial, além da necessidade de representar todas as combinações, restrições, *etc.*, é importante (senão fundamental) associar a LinExp à dados de domínio, anotações sobre atividades e informações sobre os programas utilizados, pois estas informações podem ser importantes no processo de modelagem e derivação. Além disso, a LinExp deve ser capaz de verificar se os *workflows* derivados são válidos (com ausência de *deadlocks*, *starvation*, *etc*).

Com base na necessidade semântica no relacionamento de dados nesse contexto, têm-se ontologias que permitem estruturar o conhecimento em um modelo em que se pode realizar checagens e derivações na LinExp de forma automática por meio de inferências. Dados de domínio são mapeados em um grafo que denota suas relações por meio de uma linguagem lógica sob a qual raciocinadores podem operar. Diversos domínios encontram-se mapeados em ontologias [Mcguinness et al. 2002], *e.g.*, *workflows* científicos e bioinformática. De posse de uma ontologia capaz de representar as LinExps, raciocinadores podem ser aplicados para verificar e automatizar operações em modelos concretos. Propriedades como a ausência de *deadlocks*, redundância de atividades do *workflow*, equivalência de programas que podem ser substituídos para a minimização de custos computacionais podem ser detectadas, bem como quaisquer outras passíveis de especificação na linguagem lógica adotada. Além disso, é possível assegurar a corretude da implementação em relação à especificação técnica do *workflow* e/ou prever o comportamento em caso de falhas.

Assim, esse artigo propõe uma ontologia chamada $OntoExpLine$ que combina os conceitos de LinExps, *workflows*, proveniência e dados de domínio, no processo de derivação de *workflows*. Essa abordagem integra quatro ontologias: ProvOne (Proveniência de *workflows*), EDAM (Dados de domínio da bioinformática), DCMI (metadados) e uma ontologia de LinExp para fornecer meios que auxiliem o cientista em: (i) definir atividades abstratas; (ii) oferecer apoio no processo de derivação de *workflows*; (iii) especificar e descrever dados, ferramentas e configurações; e (iv) utilizar dados de domínio na definição de atividades abstratas. A abordagem proposta nesse artigo estende [de Oliveira et al. 2012], incluindo dados de domínio e proveniência. Além desta seção, o artigo apresenta na Seção 2 o referencial teórico; na Seção 3 os trabalhos relacionados; na Seção 4 é descrita a ontologia $OntoExpLine$, suas classes e propriedades, na Seção 5 é descrito o estudo de caso realizado utilizando o *workflow* SciPhy; e por fim, na Seção 6 são apresentadas as conclusões.

2. Linha de Experimentos

A LinExp é uma abordagem para a representação de experimentos de CSE baseados em *workflows*. O processo pelo qual um *workflow* executável é obtido da LinExp é chamado de *derivação*. Um *workflow* executável é derivado da LinExp a partir da escolha de um programa para implementar cada atividade abstrata (caso seja uma atividade variante, haverá uma lista com todas as opções de programas passíveis de escolha). Além disso, as atividades escolhidas devem ser compatíveis entre si. Atividades abstratas possuem tipos, que são definidos de acordo com sua *obrigatoriedade* e *cardinalidade*. Em relação à sua cardinalidade, quando uma atividade abstrata pode ser implementada por mais de um programa, a mesma é tipada como atividade variante, *i.e.,* existe mais de uma opção de implementação para a atividade abstrata. Em relação à sua *obrigatoriedade*, quando uma atividade abstrata pode não estar presente no *workflow* executável derivado, a mesma é tipada como atividade opcional. Quando a atividade deve ser utilizada em todas as derivações possíveis, a mesma é definida como atividade mandatória. No processo de derivação de um *workflow* executável são necessários três passos básicos: definir um programa que implemente cada atividade abstrata (ou mais de um programa no caso de atividades variantes); definir quais atividades são opcionais; e por fim, relacionar atividades.

Apesar de uma LinExp ser capaz de derivar diversos *workflows* diferentes por meio das atividades variantes e opcionais, não há nenhuma associação das atividades com metadados e dados de domínio [Marinho et al. 2017]. Por exemplo, uma atividade variante da LinExp pode nos oferecer mais de uma opção de implementação, e isso pode impactar principalmente os processos de execução e análise de resultados, visto que não sabemos quais algoritmos e tipos de dados estão associados com estas variações, uma vez que esta informação não consta na estrutura da LinExp. O problema que pode ser observado é que a quantidade de metadados e dados de domínio na LinExp é limitada em sua versão atual. Estas limitações podem ser contornadas com o acoplamento das LinExps com ontologias, que fornecem um arcabouço para modelagem de conhecimento.

3. Ontologias de Apoio à Experimentos

No contexto de computação, ontologias são definidas como uma especificação formal e explícita de uma conceitualização relacionada de um domínio [Gruber et al. 1993]. Uma estrutura ontológica é composta por módulos que são constituídos por classes, e propriedades, que conectam instâncias de dado. Aplicadas no contexto de LinExp, é possível instanciar, caracterizar e relacionar programas e atividades abstratas, agregando metadados relativos à sua execução, dependências de *software*, dados do domínio de aplicação (*e.g.*, bioinformática) e recursos. A seguir discutimos algumas das ontologias usadas neste artigo para modelar a $OntoExpLine$, conforme levantamento anterior [Dias et al. 2019]. O ProvONE é um modelo de dados proposto para representar dados de proveniência [Freire et al. 2008] em *workflows*, e que possui uma ontologia associada (`https://github.com/DataONEorg/sem-prov-ontologies`). O ProvONE se baseia na recomendação do W3C PROV (`https://www.w3.org/TR/prov-primer/`). Suas classes e relações são voltadas ao relacionamento de dados de proveniência relativos à especificação do *workflow*, sua execução, dados consumidos e gerados. O desenvolvimento da ontologia foi embasado na dificuldade de interoperabilidade de dados de *workflows* por parte dos SGWfs, que em sua grande maioria utilizam formatos proprietários para representar o *workflow*. A ontologia ProvONE é dividida em três ramos principais, que integram 15 classes, e 23 propriedades. O primeiro ramo (*Trace Representation*) representa dados relacionados à execução do *workflow*. O ramo *Workflow Evolution Representation* relaciona informações sobre a especificação e evolução do *workflow*, e o ramo *Data Structure*

Representation que é voltado à caracterização das instâncias de dados.

A EDAM (`http://edamontology.org/`) é uma ontologia do domínio da biologia. Sua estrutura inclui classes sobre identificadores de dados, formatos, operações e tópicos, e se destina à organização, localização e relacionamento de ferramentas e dados que podem ser integrados ou consumidos por *workflows* complexos [Ison et al. 2013]. A EDAM é composta por 3.360 classes, onde os termos organizados por essa ontologia são conectados por 14 propriedades. Suas classes podem ser subdivididas em quatro ramos específicos: (i) *dados*: são os registro de dados que são legíveis por *software* que consomem e produzem dados; (ii) *formatos*: tipo de estruturação para representar dados em arquivos, *strings*, mensagem, ou diretórios; (iii) *operações*: funções ou operações que processam um conjunto de entradas e geram conjuntos de saídas; e (iv) *tópicos*: categorias que representam um domínio ou campo de interesse. A *Dublin Core Metadata Initiative* (`https://www.dublincore.org/specifications` — DCMI), possui a finalidade de apoiar a descoberta de recursos na *web* [Weibel and Koch 2000]. Sua estrutura permite representar metadados em RDF, JSON, XML e bancos de dados relacionais, possibilitando assim, que sejam aplicados conceitos como relações de domínio, intervalos, subpropriedades e subclasses no processo de definição de relacionamentos de metadados. Essa característica possibilita com que o cientista foque esforços no processo de descrição textual dos metadados, pois a DCMI abstrai diversas técnicas de construção de relacionamentos entre os componentes de metadados. A estrutura é composta por 15 elementos opcionais e repetíveis, utilizados para descrever metadados: título, criador, assunto, descrição, editor, colaborador, data, tipo, formato, identificador, fonte, idioma, relação, cobertura e direitos. A utilização desse modelo é vantajosa em vários aspectos, destacando-se: (i) Simplificação da reprodutibilidade, uma vez que possibilita a conservação de dados e módulos do *workflow* a longo prazo em formato estruturado de especificação não-ambígua; (ii) Validação de consistência através de inferências automatizadas; e (iii) Simplificação e/ou redução de *overhead* de processamento, uma vez que raciocinadores podem realizar inferências complexas de maneira otimizada.

4. $OntoExpLine$: uma Ontologia para Linhas de Experimentos

Diferentes tipos de ontologias podem ser desenvolvidos. As ontologias podem ser classificadas como de *Nível Superior*, *Domínio*, *Tarefa* e *Aplicação*. De acordo com [Guarino 1997], uma ontologia de nível superior descreve elementos genéricos como espaço e tempo. Uma ontologia de domínio apresenta os relacionamentos entre objetos presentes em um domínio específico. A ontologia de tarefa difere da ontologia de domínio, uma vez que descreve o vocabulário relacionado a uma tarefa por meio da especialização dos conceitos introduzidos em uma ontologia de nível superior. Uma ontologia de aplicação contém conceitos que pertencem simultaneamente a um domínio e uma tarefa. Desenvolver uma ontologia não é uma tarefa simples. Para tanto, seguimos a abordagem SABiO [de Almeida Falbo 2014]. A SABiO considera uma série de passos, a saber: (i) *Identificação do Propósito*, (ii) *Captura dos Conceitos*, (iii) *Formalização*, (iv) *Integração com outras ontologias*, (v) *Avaliação* e (vi) *Documentação*. A seguir discutimos como cada etapa foi seguida no desenvolvimento da $OntoExpLine$. A $OntoExpLine$ é uma ontologia de tarefa que faz uso de dados de domínio no processo de especificação de *workflows* abstratos e derivação de *workflows* executáveis. A primeira atividade no processo de construir uma ontologia é a Identificação do Propósito. Neste trabalho, o objetivo da ontologia é representar tarefas associadas às LinExps. Assim, foi definida a seguinte questão de competência: "Quais são os principais conceitos envolvidos no desenvolvimento de uma LinExp?" A resposta a esta pergunta é importante porque até agora, com o melhor de nosso conhecimento, não há iniciativa para organizar os conceitos de LinExps.

A Captura dos Conceitos envolve a identificação e especificação de conceitos (classes),

seus relacionamentos e todos os outros elementos que se fazem necessários para o desenvolvimento da ontologia, como axiomas, instâncias, *etc*. Os conceitos associados às LinExps foram identificados: *Linha de Experimento*, *Atividade Abstrata*, *Tipo de Atividade* (*Mandatória*, *Opcional* e *Variante*) e *Relação* (uma vez que é baseada no modelo algébrico de [Ogasawara et al. 2011]). Uma LinExp é composta de atividades abstratas, onde cada atividade está associada a um tipo. Atividades abstratas consomem e produzem relações (associadas a um *schema* com os atributos). Após capturar os conceitos da $OntoExpLine$, sua Formalização teve início. Várias linguagens podem ser usadas para esse fim, como a OWL (*Web Ontology Language*), linguagem escolhida nesse trabalho. Conforme apresentado na Figura 1, a ontologia desenvolvida conecta módulos que relacionam dados da LinExp, do *workflow* executável derivado, de dados de domínio e metadados genéricos. O módulo de LinExp (classes em cinza) é utilizado para criar atividades abstratas de uma LinExp e definir seus respectivos tipos. Este ramo é conectado à classe *Program* (implementa atividades abstratas), e *Metadata*. O bloco relacionado aos metadados genéricos por sua vez, é conectado as classes *Program* (do módulo de proveniência prospectiva, retrospectiva, e de processo - em azul e amarelo, respectivamente), e a classe *Activity Type* do módulo de LinExp. Esse relacionamento é necessário para que anotações relacionadas à execuções e configurações sejam agregadas aos itens que compõem o *workflow* executável derivado. De maneira análoga ao módulo de dados de domínio, o módulo de metadados genéricos também pode ser alterado de acordo com a necessidade do cientista, visto que as classes que o compõem possuem perfil opcional. É importante ressaltar que o ramo relativo aos dados de domínio, estendido da ontologia EDAM, tem o papel de agregar conceitos de uma esfera de conhecimento específica, visando especificação das atividades que compõem o experimento por meio de atividades abstratas. Com base nisso, caso seja necessário aplicar a $OntoExpLine$ à outros domínios (*e.g.*, astronomia), tal ramo deve ser alterado para considerar os conceitos do novo domínio.

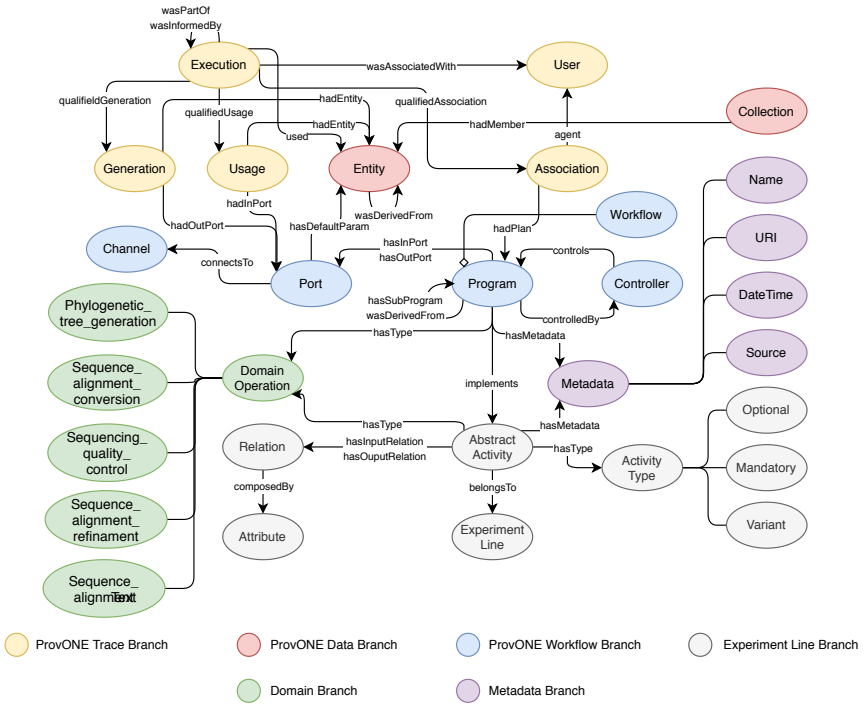

Figura 1. Estrutura da ontologia $OntoExpLine$.

5. Avaliação da $OntoExpLine$

De forma a realizar a Avaliação da ontologia proposta, a $OntoExpLine$ foi avaliada considerando seus requisitos e corretude. Esse processo foi realizada em duas etapas: (i) avaliação por um especialista na área e (ii) estudo de viabilidade com um experimento real. Para o primeiro passo, a $OntoExpLine$ foi avaliada por dois pesquisadores envolvidos na redação deste artigo, mas diferentes do responsável pela definição da ontologia. Para cada conceito presente na $OntoExpLine$ foi solicitado que o pesquisador o avaliasse considerando o seguintes resultados possíveis: (i) *Conformidade Total* (CT), (ii) *Conformidade Parcial* (CP) e (iii) *Não Conforme* (NC). Caso a resposta fosse CP ou NC, o pesquisador deveria justificá-la. Os critétios de avaliação sugeridos pela SABiO são: (i) Clareza, (ii) Coerência, (iii) Extensibilidade, e (iv) Viés de Codificação Mínimo (uso de símbolos específicos). Em geral, pequenos ajustes foram necessários para esclarecer as diferenças entre alguns conceitos de LinExp. Apenas o critério *Clareza* não foi definido como *Conformidade Total* pelos avaliadores. A explicação para isso foi que foram utilizados axiomas para representar restrições (*e.g.*, um *Programa* que implementa o conceito de *Geração de Árvore Filogenética* não pode estar associado a uma atividade abstrata que implementa um conceito diferente), o que faz com que não seja tão clara a visualização da $OntoExpLine$.

Para testar a viabilidade da $OntoExpLine$ com um estudo de caso real, escolhemos o *workflow* SciPhy [Ocaña et al. 2011], do domínio da bioinformática. O SciPhy tem como objetivo construir árvores filogenéticas a partir de sequências de DNA, RNA e aminoácidos. Seu *dataflow* é composto por cinco atividades: (i) normalização dos dados de entrada; (ii) alinhamento de sequências; (iii) conversão de formatos para processamento posterior; (iv) escolha do modelo evolutivo correspondente à sequência de entrada; e finalmente (v) geração das árvores filogenéticas. No SciPhy diferentes programas podem implementar uma mesma atividade. Isso possibilita ao cientista avaliar diferentes métodos e algoritmos. Vale salientar também que, cada um desses programas consome diferentes níveis de recursos computacionais (como memória e CPU), o que pode influenciar por exemplo, no custo de execução do experimento.

Dessa forma, percebe-se que o conceito de LinExp é adequado para modelar o SciPhy, pois o mesmo pode ser executado por diferentes conjuntos de programas, gerando assim, diferentes produtos de dados durante sua execução. Assim, o SciPhy foi mapeado e estruturado na ontologia, e seu esquema é representado na Figura 2. O primeiro passo para a estruturação da LinExp do SciPhy de modo que fossem identificados pontos de derivação, foi levantar suas atividades abstratas (representados na Figura 2 pelos módulos branco e cinza), e quais programas podem implementá-las (retângulos verdes). Essa tarefa foi realizada levando em consideração todas as variações possíveis do *workflow* e os programas relacionados no ramo de domínio estendido da ontologia de domínio EDAM, tarefa essa que contou com o auxílio de um especialista de domínio. Identificados os programas que implementam atividades abstratas, foram definidas portas/atributos de entrada (círculos em amarelo) e saída (círculos azuis) para cada programa relacionado.

As portas/atributos de entrada e saída utilizadas pelos programas são conectadas por relações (setas), e cada transição de atividade abstrata é representada por um canal (duto tracejado), que encapsula relações. A visualização de pontos variantes foi identificada em duas atividades abstratas, representadas pelos retângulos em cinza (atividades abstratas 2 e 5). Os atributos também são representados na ontologia e os mesmos compõem coleções consumidas e produzidas . No esquema, a atividade 4 gera a coleção **ModelGeneratorOutputDataset** composta pelo atributo *EvolutiveModel*. O *dataset* gerado pela atividade 4 pode ser consumido por dois programas implementadores da atividade 5. Entretanto, após consumir o mesmo conjunto

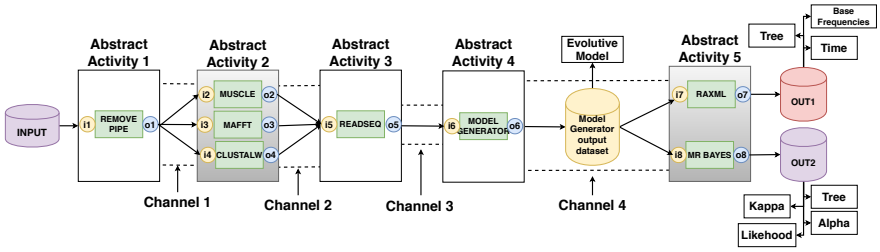

Figura 2. Estudo de caso realizado com o *workflow* SciPhy.

de dados, os programas implementadores variantes geram *datasets* compostos por atributos diferentes: enquanto **OUT1** é composto por *Tree*, *BaseFrequencies*, e *Time*; **OUT2** é composto por *Tree*, *Kappa*, *Alpha* e *Likehood*. Com base nisso, percebe-se que no contexto de experimentos baseados em *workflows*, fatores estruturais da fase de composição podem interferir em fases posteriores, como por exemplo nas fases de execução e análise dos dados. Isso porque os produtos de dados gerados durante a execução do experimento podem variar de acordo com o ferramental empregado, o que pode impactar em questões como viabilidade de execução e análise de dados. Os *workflows* executáveis derivados foram comparados com resultados anteriores[1] do grupo e foi constatado que todas as variações do SciPhy que foram derivadas se encontram corretas e equivalentes as especificadas de forma manual. Toda a Documentação da *OntoExpLine* pode ser obtida em sua página no GitHub[2].

6. Conclusão

As LinExps auxiliam na execução de experimentos de CSE oferecendo apoio ao cientista em processos de especificação, montagem e validação de experimentos deriváveis, ou seja, que são compostos por atividades abstratas variantes que podem ser implementadas por diferentes programas. A derivação de *workflows* executáveis por uma representação de mais alto nível é a principal vantagem das LinExps. Apesar destas representarem uma contribuição e um avanço na gerência de experimentos de CSE, apresentam limitações que tangem à semântica envolvida.

O processo de definição de uma LinExp não é trivial, pois demanda definições, conexões de componentes heterogêneos e verificações de diversas naturezas, para fins de checagem de corretude em relação às especificações e viabilidade técnica dos *workflows* derivados. Assim, faz-se necessário o uso de mecanismos que capazes de executar validações das composições geradas de forma automática, a fim de evitar características indesejadas como *deadlock* e *overhead*. Ontologias aplicadas neste contexto podem oferecer apoio não apenas ao processo de derivação e certificação de *workflows* a partir de LinExps, mas também na fase de análise dos dados. Isso porque além dos componentes da LinExp, podem ser agregados dados de proveniência e de domínio à essas ontologias. Isso corrobora, inclusive, com a reprodutibilidade de experimentos.

Este artigo apresentou a $OntoExpLine$, uma ontologia de LinExp que pode ser usada para apoiar o processo de experimentação científica. A $OntoExpLine$ foi desenvolvida seguindo a abordagem SABiO de desenvolvimento de ontologias. Desta maneira, o cientista pode modelar seu *workflow* em níveis mais altos de abstração, sem se preocupar com aspectos de infraestrutura, e utilizar mecanismos de inferência/raciocinadores para produzir de forma automática versões variantes validadas de um mesmo *workflow*. Uma representação do *workflow* com semântica, como é o caso da linha de experimento aliada com ontologias, é particularmente

[1] https://bitbucket.org/vitorss/sciphy/src

[2] `https://github.com/UFFeScience/OntoExpLine`

importante para experimentos em larga escala. Para avaliar a proposta, utilizamos um *workflow* real da bioinformática (SciPhy), que já se encontra especificado em diferentes executáveis no SGWf SciCumulus em trabalhos anteriores do grupo de pesquisa. Aplicando a $OntoExpLine$ no contexto do SciPhy, pudemos comparar os resultados obtidos através de sua derivação via *OntoExpLine* com as especificações anteriores do SciPhy, corroborando a corretude das derivações do SciPhy, obtidas após seu mapeamento na OntoExpLine.

Referências

Carvalho, L. A. M. C., Garijo, D., Medeiros, C. B., and Gil, Y. (2018). Semantic software metadata for workflow exploration and evolution. In *IEEE eScience*, pages 431–441. IEEE Computer Society.

de Almeida Falbo, R. (2014). Sabio: Systematic approach for building ontologies. In *Joint Workshop ONTO.COM / ODISE on Ontologies in Conceptual Modeling and Information Systems Engineering*, CEUR Workshop Proceedings. CEUR-WS.org.

de Oliveira, D., Ogasawara, E. S., Dias, J., Baião, F. A., and Mattoso, M. (2012). Ontology-based semi-automatic workflow composition. *JIDM*, 3(1):61–72.

Dias, L. G., Lopes, B., and de Oliveira, D. (2019). Aplicação de ontologias de proveniência em workflows científicos: um mapeamento sistemático. In *XIII BreSci*. SBC.

Freire, J., Koop, D., Santos, E., and Silva, C. T. (2008). Provenance for computational tasks: A survey. *Comput. Sci. Eng.*, 10(3):11–21.

Gil, Y., Ratnakar, V., Kim, J., González-Calero, P. A., Groth, P. T., Moody, J., and Deelman, E. (2011). Wings: Intelligent workflow-based design of computational experiments. *IEEE Intell. Syst.*, 26(1):62–72.

Gruber, T. R. et al. (1993). A translation approach to portable ontology specifications. *Knowledge acquisition*, 5(2):199–221.

Guarino, N. (1997). Understanding, building and using ontologies. *International Journal of Human-Computer Studies*, 46(2-3):293–310.

Ison, J., Kalaš, M., Jonassen, I., Bolser, D., Uludag, M., McWilliam, H., Malone, J., Lopez, R., Pettifer, S., and Rice, P. (2013). Edam: an ontology of bioinformatics operations, types of data and identifiers, topics and formats. *Bioinformatics*, 29(10):1325–1332.

Marinho, A., de Oliveira, D., Ogasawara, E. S., Sousa, V. S., Ocaña, K. A. C. S., Murta, L., Braganholo, V., and Mattoso, M. (2017). Deriving scientific workflows from algebraic experiment lines: A practical approach. *FGCS*, 68:111–127.

Mcguinness, D. L., Fikes, R., Hendler, J., and Stein, L. A. (2002). Daml+oil: an ontology language for the semantic web. *IEEE Intelligent Systems*, 17(5):72–80.

Ocaña, K. A., de Oliveira, D., Ogasawara, E., Dávila, A. M., Lima, A. A., and Mattoso, M. (2011). Sciphy: a cloud-based workflow for phylogenetic analysis of drug targets in protozoan genomes. In *BSB*, pages 66–70. Springer.

Ogasawara, E. S., de Oliveira, D., Valduriez, P., Dias, J., Porto, F., and Mattoso, M. (2011). An algebraic approach for data-centric scientific workflows. *PVLDB*, 4(12):1328–1339.

Sameh, A., Cybenko, G., Kalos, M., Neves, K., Rice, J., Sorensen, D., and Sullivan, F. (1996). Computational science and engineering. *ACM Comput. Surv.*, 28(4):810–817.

Weibel, S. L. and Koch, T. (2000). The dublin core metadata initiative. *D-lib magazine*, 6(12):1082–9873.