REPRODUÇÃO DE EXPERIMENTOS COMPUTACIONAIS NA INFRAESTRUTURA DE COMPUTAÇÃO EM NUVEM

Ary Henrique Morais de Oliveira

Tese de Doutorado apresentada ao Programa de Pós-graduação em Engenharia de Sistemas e Computação, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Doutor em Engenharia de Sistemas e Computação.

Orientadores: Marta Lima de Queirós Mattoso
Daniel Cardoso Moraes de Oliveira

Rio de Janeiro
Setembro de 2015

REPRODUÇÃO DE EXPERIMENTOS COMPUTACIONAIS NA INFRAESTRUTURA DE COMPUTAÇÃO EM NUVEM

Ary Henrique Morais de Oliveira

TESE SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA (COPPE) DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE DOUTOR EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Examinada por:

Prof.a Marta Lima de Queirós Mattoso, D.Sc.

Prof. Daniel Cardoso Moraes de Oliveira, D.Sc.

Prof. Fabio Andre Machado Porto, D.Sc.

Prof.a Fernanda Araújo Baião Amorim, D.Sc.

Prof. Leonardo Gresta Paulino Murta, D.Sc.

Prof. Geraldo Bonorino Xexéo, D.Sc.

RIO DE JANEIRO, RJ – BRASIL
SETEMBRO DE 2015

Oliveira, Ary Henrique Morais de

Reprodução de Experimentos Computacionais na Infraestrutura de Computação em Nuvem/Ary Henrique Morais de Oliveira. – Rio de Janeiro: UFRJ/COPPE, 2015.

XVIII, 181 p.: il.; 29, 7cm.

Orientadores: Marta Lima de Queirós Mattoso

Daniel Cardoso Moraes de Oliveira

Tese (doutorado) – UFRJ/COPPE/Programa de Engenharia de Sistemas e Computação, 2015.

Referências Bibliográficas: p. 151 – 164.

1. *Workflows* Científicos. 2. Reprodutibilidade Científica. 3. Proveniência. 4. Computação em Nuvem. I. Lima de Queirós Mattoso, Marta *et al.* II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia de Sistemas e Computação. III. Título.

Aos meus pais Ary e Ana Tereza
e irmãos Bruno e Ivo.

Agradecimentos

O desenvolvimento de uma tese de doutorado acontece de forma concomitante a diversos contextos e situações boas e ruins em nossa vida. Esse período de quatro anos traz diversas situações diferentes e algumas vezes inusitadas. Situações que apresentam-se na forma de desafios que muitas vezes nos testam diversas aspectos, seja intelectualmente, ou em termos de paciência, obstinação, perseverança, e assim por diante, e que muitas vezes colocam em cheque a nossa capacidade de superação.

Claro que nesses momentos nos agarramos a toda a força que ainda temos, e buscamos sentimentos diversos para superar esses desafios. Buscamos o conforto da família quando nos sentimos cansados, a paz dos nossos lares quando estamos fracos, a inspiração dos nossos orientadores quando estamos angustiados e com medo, e até mesmo, o incentivo que recebemos dos nossos amigos, alunos e pessoas especiais que nos cercam. Inspirado nesses sentimentos, eu gostaria de agradecer as pessoas e instituições que me ajudaram a chegar a este momento, no qual eu jamais tinha imaginado antes do ano de 2.010, quanto ingressei no programa de doutorado.

Primeiramente agradeço à Deus por todos as oportunidades e graças que me foram concedidas, independente dos resultados, acredito que as oportunidades são as grandes dádivas divinas, e este doutorado com certeza foi uma das maiores graças que eu recebi na minha vida.

Aos meus pais que sempre deram o seu apoio me incentivando a perseguir com determinação meus objetivos, independente das dificuldades e obstáculos. Agradeço o exemplo de carinho, paciência e benevolência do meu pai, Ary Oliveira, ao exemplo de força, garra e persistência da minha mãe Ana Tereza.

Aos meus irmãos por todo o apoio e carinho, com certeza as mensagens e ligações de apoio vinham como um combustível para continuar a luta. Aos meus sobrinhos, que tiveram paciência para entender quando o tio não pudia dar a atenção que eles mereciam. A minha noiva Glenda Michele pelo apoio nos momentos de dúvida e medo, por toda a compreensão e paciência nos momentos de angústia e ausência, e por todo o carinho nessa reta final.

À professora Marta Lima de Queirós Mattoso, um exemplo de professora, profissional, enfim, de pesquisadora. Agradeço todo o conhecimento, atenção, apoio e incentivo. Com certeza eu não teria palavras suficientes para expressar o quanto sou

grato pela oportunidade de trabalhar sob sua orientação. Ao professor Daniel de Oliveira, o qual admiro pela dedicação, atenção, e sem dúvidas pelas contribuições que foram essenciais para o desenvolvimento deste trabalho.

Aos amigos Rogério Borba, Glendara Martins, Rafael Carvalho, Warley Gramacho por todo apoio e incentivo. Aos meus alunos orientados do Curso de Ciência da Computação da Universidade Federal do Tocantins, pelo auxílio e paciência durante esse período, em especial: Igor Barreto, Igor Modesto, Murilo Martins, Maycon Junqueira, Dábila Cristina, Marcelo Cláudio, Thaylon Guedes.

À Kary Ocaña e Vitor Silva Sousa que compartilharam seu conhecimento e experiência com diversos elementos que usei nesta tese e que foram fundamentais para a preparação dos experimentos e concepção dos resultados.

Aos colegas do colegiado do Curso de Ciência da Computação da Universidade Federal do Tocantins pelo apoio durante os períodos de ausência, por aceitarem readequações de horários, acreditando e apoiando este programa de capacitação.

Ao colegiado do Programa de Engenharia de Sistemas e Computação PESC/-COPPE/UFRJ por ter oportunizado o doutorado para nosso corpo docente do Curso de Ciência da Computação da Universidade Federa do Tocantins. Com certeza com a distância e dificuldades em nossa região, dificilmente conseguiríamos afastamentos a longo prazo para a qualificação de todo corpo docente nesse curto espaço de tempo. Portanto, gostaria de deixar um agradecimento especial ao prof. Geraldo Xexéo que abraçou a causa e nos deu todo o suporte viabilizando o programa DINTER/UFT.

Agradeço a todos.

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Doutor em Ciências (D.Sc.)

REPRODUÇÃO DE EXPERIMENTOS COMPUTACIONAIS NA INFRAESTRUTURA DE COMPUTAÇÃO EM NUVEM

Ary Henrique Morais de Oliveira

Setembro/2015

Orientadores: Marta Lima de Queirós Mattoso
Daniel Cardoso Moraes de Oliveira

Programa: Engenharia de Sistemas e Computação

Workflows científicos são utilizados para modelar experimentos computacionais. Os resultados desses experimentos são publicados e compartilhados na forma de artigos apresentados nos veículos científicos. Entretanto, para que tais resultados sejam cientificamente válidos eles devem ser passíveis de reprodução. Pesquisadores têm a necessidade de compartilhar os artefatos utilizados para a geração dos resultados, dentre eles, os dados de entrada do *workflow* e os parâmetros utilizados no experimento. Entretanto, reproduzir um experimento baseado nestes artefatos não é uma tarefa trivial. Apesar de o *workflow* especificar o protocolo de execução com os artefatos disponíveis, nem sempre o ambiente de execução da reprodução pode ser compatível. Programas e bibliotecas que foram originalmente utilizados podem estar obsoletos e incompativeis. Esse cenário se torna ainda mais complexo quando tratamos de reproduções de longo prazo, como por exemplo, diversos anos após a execução que levou aos resultados publicados. Diante disto, esta tese propõe o desenvolvimento da abordagem ReproeScience para tratar a reprodução de experimentos modelados com *workflows*. São usadas as tecnologia de máquinas virtuais e computação em nuvem para auxiliar na reprodução do ambiente onde o experimento foi originalmente executado. O objetivo é preparar o ambiente de execução de forma que ele possa ser instanciado sob demanda e reproduzido em condições equivalentes na nuvem. Para mostrar a efetividade da abordagem foram executados dois estudos de caso com *workflows* da astronomia e bioinformática. Os resultados dos experimentos de ambos estudos de casos confirmaram a sua reprodução, e ainda mostraram algumas peculiaridades a serem tratadas para a concepção de mecanismos reprodução, bem como para a verificação e validação dos resultados.

Abstract of Thesis presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Doctor of Science (D.Sc.)

COMPUTATIONAL EXPERIMENTS REPRODUCTION IN THE CLOUD COMPUTING INFRASTRUCTURE

Ary Henrique Morais de Oliveira

September/2015

Advisors: Marta Lima de Queirós Mattoso
Daniel Cardoso Moraes de Oliveira

Department: Systems Engineering and Computer Science

Scientific workflows are used to model computational experiments. The results of these experiments are published and shared in the form of articles presented at scientific vehicles. However, for these results to be scientifically valid, they must be reproducible. Researchers have the need to share the artifacts used for results generation, for example workflow input data and parameters used in the experiment. However, to reproduce an experiment based on these artifacts is not a trivial task. Although the workflow specifies the execution protocol with the available artifacts, the reproduction execution environment may not always be compatible. Programs and libraries that were originally used may be obsolete and incompatible. This situation becomes even more complex when dealing with long-term reproduction, for example, several years after the execution that led to the published results. Then, this thesis proposes the development of ReproeScience approach to deal with reproduction of experiments modeled with workflows. Virtual machines technology and cloud computing are used to assist in the reproduction of the environment where the experiment was originally performed. The goal is to prepare the execution environment so that it can be instantiated on demand and reproduced under equivalent conditions in the cloud. To show the effectiveness of the approach two case were performed studies with astronomy and bioinformatics workflows. The experiments results of both studies confirmed the case studies reproduction and also showed some peculiarities that will be treated for the design of reproduction mechanisms and results validation and verification.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Algoritmos

Lista de Abreviaturas

ACM	*Association for Computing Machinery*, p. 11
AMI	*Amazon Machine Image*, p. 105
APS	*American Physics Society*, p. 10
CDE	*Code, Data and Environment*, p. 93
EC2	*Elastic Cloud Compute*, p. 105
EMI	*Eucalyptus Machine Image*, p. 105
HPC	High-Performace Computing, p. 1
ICERM	*Institute for Computational and Experimental Research in Mathematics*, p. 12
IODA	*Interactive Open Document Architecture*, p. 100
NIST	*National Institute of Standards and Technology*, p. 32
OPM	*Open Provenance Model*, p. 2
PASS	*Provenance Aware Storage Systems*, p. 92
PDF	*Portable Document Format*, p. 102
RR	*Reproductible Research*, p. 6
SCons	*Software Construction*, p. 102
SGWfC	Sistema de Gerência de *Workflow* Científico, p. 2
SHARE	*Sharing Hosted Autonomous Research Environments*, p. 97
SIGMOD	*Special Interest Group on Management of Data*, p. 11
SLA	*Service Level Agreenment*, p. 3
TIC	Tecnologia da Informação e Comunicação, p. 1

URL	*Uniform Resource Locator*, p. 51
VMI	*Virtual Machine Image*, p. 4
VM	*Virtual Machine*, p. 4
W3C	*Wide World Web Consortium*, p. 2
WFA	*Workflow Agent*, p. 106
WSSE	*Whole System Snapshot Exchange*, p. 104

Capítulo 1

Introdução

O uso de sistemas de computadores têm acelerado o desenvolvimento da ciência, aumentado a eficiência e a eficácia no processo de análise e produção de dados em experimentos científicos. Tais experimentos fazem o uso de instrumentos/sensores e são baseados em simulações, produzindo um grande conjunto de dados [13–16], o que caracteriza a ciência como intensiva de dados. Neste cenário, a computação estabeleceu-se como o terceiro eixo da ciência, ao lado das ciências experimental e teórica [17]. A computação auxilia a ciência teórica com a implementação e simulação de modelos complexos, em ambientes e situações hipotéticas, e apoia a ciência experimental na coleta, armazenamento e análise das informações obtidas através de sensores e instrumentos especializados para o monitoramento e registro de fenômenos nos ambientes nos quais estão inseridos.

Ambas as ciências estão produzindo grandes volumes de dados espalhados por inúmeras quantidade de arquivos muitas vezes organizados em uma estrutura complexa [18]. Diante disso, surgiu a *e-Science* com o desafio de prover mecanismos para processar e analisar esses grandes conjuntos de dados em projetos científicos. A *e-Science* é uma ciência em larga escala realizada através de colaborações globalmente distribuídas, possíveis através da *internet* [18]. Uma das características dessas colaborações é a necessidade de acesso a grandes coleções de dados, recursos de computação em larga escala e visualização de análise de resultados de computação de alto desempenho. A *e-Science* emprega métodos e ferramentas de Tecnologia da Informação e Comunicação (TIC) nos diversos campos do conhecimento para a coleta, gerenciamento, análise, visualização e compartilhamento de dados científicos, apoiando todas as atividades do ciclo de vida da ciência para torná-la mais eficiente e produtiva.

Muitas análises científicas são executadas por um conjunto de recursos especiais de computação, tais como *hardware* especializado, *software* e bibliotecas preparados para execução de atividades complexas, que muitas vezes envolvem computação de alto desempenho (*High-Performace Computing* - HPC) e a manipulação de grandes

conjuntos de dados [19]. Em geral, os experimentos computacionais são compostos por um conjunto de atividades complexas, executadas por programas de computador que são encadeadas e orquestradas de forma que a saída de uma atividade é consumida como a entrada da próxima atividade no fluxo. Tal encadeamento pode gerar um grande volume de dados intermediários e, portanto, demanda métodos automatizados de gerência e análise desse conjunto de informações.

Diante desse contexto, foram concebidas as ferramentas de *Workflows* Científicos como uma abordagem bastante promissora no processo de gerência das atividades e produção dos dados gerados pela execução dos experimentos computacionais. Um *workflow* é uma abstração que permite a composição de um conjunto de programas e *scripts* na forma de fluxo de atividades com o objetivo de atingir um resultado desejado [20]. Ele é composto por um conjunto de tarefas e dependências de dados entre elas, onde uma tarefa posterior aguarda um conjunto de dados produzido por uma tarefa anterior no fluxo de dados. Os *workflows* são modelados, projetados e executados por Sistemas de Gerência de *Workflows* Científicos (SGWfC).

Os SGWfCs são sistemas de *software* que permitem aos cientistas definir, implementar e monitorar a execução dos seus experimentos por meio de um *workflow* científico. Os SGWfCs também são responsáveis por coletar, armazenar e gerenciar os metadados e dados de proveniência da execução dos *workflows*, incluindo todas as informações sobre os agentes (pessoas, organização, *softwares*), atividades (*softwares*, bibliotecas, *scripts*) e entidades (conjunto de dados, parâmetros) envolvidos na produção de um pedaço de dados ou objeto. Esse conjunto de metadados e proveniência são usados para formar as avaliações sobre a qualidade, confiabilidade e confiança dos resultados produzidos por um *workflow* [8].

Os metadados e dados de proveniência auxiliam no processo de publicação, compartilhamento e reprodução de um experimento, bem como na reutilização de partes do processo e do conhecimento adquirido. Isto permite que o protocolo científico e os resultados produzidos por um experimento sejam publicados e compartilhados. Além disso, a reprodução ocorre a partir de um ponto consolidado da pesquisa com acesso a todos os objetos usados na produção dos resultados. Esta característica elimina a necessidade de reconstrução do experimento a partir do ponto inicial, dando acesso a informações suficientes para subsidiar a continuidade de novas pesquisa a partir do conhecimento consolidado. Consequentemente, a ciência ganha um aumento na velocidade da disseminação das informações.

Os *workflows* científicos juntamente com os dados e informações sobre o desenvolvimento de uma Pesquisa Reprodutível (*Reproducible Research* - RR) ajudam a descrever os resultados e o protocolo científico, auxiliando na repetição e reprodução de um experimento [21]. Goble [22] enfatiza a importância da pesquisa reprodutível definindo que a reprodução é o princípio do método científico, que auxilia no processo

de comparação e revisão de métodos empregados e resultados alcançados, verificando se os resultados estão corretos, e se o método é convincente e repetível. Além disso, para que um experimento seja válido sob o ponto de vista científico, o seu resultado deve ser passível de reprodução por terceiros.

A capacidade de reprodução dos experimentos científicos não envolve apenas a reprodução da metodologia, do método e dos resultados finais, mas também de todos os artefatos e elementos usados para a derivação dos resultados. Porém, para permitir a reprodução não basta apenas ter o protocolo seguido para a execução do experimento, bem como os dados de proveniência, é necessário ainda prover ao cientista o acesso ao conjunto de dados, infraestrutura, plataforma de *software*, parâmetros de configuração necessários para a execução de um experimento de uma forma simples e conveniente, com mínimo esforço possível e com poder computacional necessário [23]. Uma das principais preocupações em termos de reprodução de experimentos é a questão da rápida evolução da infraestrutura dos sistemas de computação. Dada esta evolução, ainda será possível recuperar a infraestrutura e garantir a reprodução de um experimento concebido em uma tecnologia obsoleta?

A publicação do *workflow*, juntamente com os dados de proveniência, incorporados às produções científicas permitem o aumento da produtividade do trabalho dos cientistas. Adicionando tais características, a possibilidade de reproduzir o experimento tendo acesso a todos os elementos originalmente utilizados na execução e, tendo a capacidade de modificação desses experimentos, pode tornar o desempenho do trabalho científico ainda mais eficiente e eficaz. Entretanto, é importante observar que muitos experimentos computacionais demandam por recursos de alto poder computacional e ambientes complexos. Em muitos casos, o cientista que deseja reproduzir este tipo de experimento não tem acesso a tais recursos e não pode adquirir infraestrutura computacional com elevado custo financeiro apenas para executar essa reprodução. Portanto, é necessário buscar formas de fornecer um ambiente computacional para que os interessados na reprodução possam obter, implantar e reproduzir uma pesquisa de forma a verificar os resultados e, a partir disto, contribuir com o progresso da ciência continuando a pesquisa de um ponto mais consolidado.

A computação em nuvem pode trazer diversas vantagens neste e nos mais variados cenários para comunidade científica [24], pois trata-se de um paradigma de fornecimento de serviços de infraestrutura de *data center*, *hardware* e *software*, através de uma rede de computadores. A computação em nuvem, segundo a definição do *National Institute of Standards and Technology* (NIST) [25], é um modelo que permite o acesso onipresente, conveniente e sob demanda a um conjunto de recursos computacionais configuráveis através de uma rede de computadores, permitindo que tais recursos possam ser rapidamente fornecidos e liberados com o mínimo esforço de gerenciamento e interação com o fornecedor do serviço. Este modelo foi concebido

para fornecer recursos de forma flexível, altamente escalável e sob demanda.

A computação em nuvem é fundamentada em quatro tecnologias [24, 26, 27]: grade computacional, computação de utilidade, virtualização de recursos e arquitetura orientada a serviços. A grade computacional é uma forma de computação distribuída que permite que diversos recursos de computadores heterogêneos conectados à rede trabalhem no processamento de uma tarefa complexa ao mesmo tempo [27]. A computação de utilidade é um modelo de fornecimento em que um fornecedor disponibiliza os recursos de computação na forma de um serviço. Segundo Kearney [28], este fornecimento adota alguma métrica clara, tal como quantidade de recursos adquiridos, perfil dos recursos, período de utilização, serviço embarcados (*backup*, alta disponibilidade e etc). A virtualização do ambiente de computação permite que vários recursos de *hardware* e *software* sejam gerenciados na forma de um *pool* de recursos. Na virtualização um computador físico pode acomodar diversos computadores lógicos, denominados máquinas virtuais [29]. A arquitetura orientada a serviços (*Service Oriented Architecture* - SOA) usa os serviços disponíveis na rede para oferecer um conjunto específico de funções. Os recursos são empacotados como serviços bem definidos que fornecem um conjunto de funcionalidades padronizadas independentes de outros serviços [30].

Portanto, a abordagem proposta nesta tese é baseada na hipótese de que é possível reproduzir o ambiente de execução do experimento e, consequentemente, dos resultados obtidos utilizando a tecnologia de Máquinas Virtuais (*Virtual Machines*-VM) para atingir a reprodução de cada um dos sistemas de computação envolvidos na execução de um *workflow*. Para isso, é proposto que o ambiente do experimento original seja transformado em Imagens de Máquinas Virtuais (*Virtual Machine Images* - VMI) para que possam ser publicadas e compartilhadas em um provedor de nuvem pública. Esta abordagem permite que interessados na reprodução tenham acesso aos recursos especiais de computação para que possam reproduzir o experimento na infraestrutura dos provedores de nuvem.

Um dos desafios para a reprodução concentra-se em capturar as configurações dos mais diversos tipos de ambientes de forma a empacotá-las em uma VMI na infraestrutura dos provedores de computação em nuvem, para que possam ser instanciadas e usadas a posteriori por terceiros. Portanto, esta tese propõe uma abordagem denominada ReproeScience (*Reproductible e-Science*) para a reprodução de experimentos computacionais modelados na forma de *workflows* científicos, com o suporte de SGWfCs como o VisTrails [31] e o SciCumulus [32], para auxiliar o cientista na composição, execução e posterior reprodução da infraestrutura de uma pesquisa. Essa cenário permite a construção de um artigo científico digital vinculado com a infraestrutura usada para a produção dos resultados de uma pesquisa, apresentando todo histórico de derivação dos experimentos até a produção do resultado final por

meio dos metadados e dados de proveniência, criando meios para verificar e validar os resultados.

A abordagem ReproeScience foi concebida para atender a um conjunto de variáveis relacionadas com o nível de reprodução de uma pesquisa, tais como a portabilidade do experimento de um ambiente operacional de computação de alto desempenho para a infraestrutura de provedores de nuvem, analisando o nível de transparência de uma pesquisa, tendo como meta um maior nível de compartilhamento dos artefatos usados para a produção dos resultados para o público em geral, e ao final, avaliar o quanto é possível reproduzir cada um desses elementos de uma pesquisa, ou seja, qual a cobertura de reprodução de umexperimento. A abordagem deve atender ainda fatores relacionados a viabilidade da movimentação dos dados para os provedores de nuvem, mesmo alguns provedores como a *Amazon Web Service*[1] optando pelo armazenamento de grandes bases de dados científicas, e ainda a características relativas ao desempenho da reprodução na nuvem em relação a execução no ambiente original e os custos financeiros pela alocação dos recursos.

A infraestrutura da nuvem traz a possibilidade de acesso a recursos especiais, tais como: ambiente de computação de alto desempenho e unidades de processamento gráfico. Além disso, permite armazenar as VMIs, tratando-as como uma unidade de compartilhamento da pesquisa. O armazenamento da infraestrutura na nuvem permite ainda manter o repositório da pesquisa eliminando a necessidade de tarefas de manutenção de um centro de dados, pois, os acordos de serviços englobam fatores de qualidade de serviço (*Quality of Service* - QoS) , tais como disponibilidade, controle de falhas da estrutura, serviços de *backup*.

1.1 Justificativa

Diversas abordagens propõem o uso da tecnologia de VMs para preservar e compartilhar o ambiente computacional. Mas ter o ambiente encapsulado em uma VM não garante a possibilidade de reprodução. Muitas pesquisas possuem características especiais, como grande volume de dados e informações para obterem inferências e provar hipóteses. Nesses casos, é necessário ter acesso a um conjunto de recursos especiais, tais como *clusters* ou grades computacionais para prover poder de processamento. Tais ambientes possuem características de serem paralelos e distribuídos, e demandam por um número expressivo de processadores para realizar o processamento. Desta forma, mesmo tendo acesso as VMs utilizadas na execução de um experimento, se o usuário não tiver acesso a essa infraestrutura de recursos de computação, ele não terá a capacidade de reproduzir o experimento.

Para ter acesso a recursos de computação de alto desempenho é necessário em um

[1]https://aws.amazon.com/datasets/

primeiro cenário implantar um *data center*, o que demanda um alto investimento financeiro para adquirir, gerenciar e manter a infraestrutura. Em um segundo cenário, pode-se tentar obter recursos em uma entidade que possui a infraestrutura necessária para a reprodução, tais como universidades e institutos de pesquisa. Porém, tais recursos são limitados a um dado número de usuários e pode requerer um processo burocrático para permitir o acesso, tornando o processo moroso. Já um terceiro cenário, proposto nesta tese, é por meio da obtenção do serviço de infraestrutura de um provedor de nuvem. Este cenário é bastante atraente, principalmente pela possibilidade de se adquirir horas de recursos de computação em empresas especializadas em fornecer a infraestrutura de *data center* na forma de serviço.

Neste último cenário, o cientista pode aumentar e reduzir a alocação de recursos baseado em sua demanda. Portanto, para reproduzir o *workflow* científico na nuvem é preciso selecionar um provedor de nuvem, em seguida fazer os devidos cadastros, que em geral incluem a sua identificação e dados financeiros e, ao final, selecionar os recursos necessários. A partir deste ponto o experimento pode ser reproduzido. Ao final da reprodução serão cobrados apenas a quantidade de recursos em relação ao tempo em que eles foram usados. Os provedores de nuvem disponibilizam bibliotecas e protocolos de comunicação que permitem criar programas que automatizam tarefas de criação e gerenciamento dos recursos ofertados. A abordagem proposta emprega tais recursos para automatizar a publicação e compartilhamento de uma pesquisa reprodutível e utiliza os recursos para permitir a reimplantação automática da infraestrutura do experimento para a reprodução da pesquisa.

1.2 Definição do Problema

Em geral, o progresso na ciência é dificultado pela incapacidade da reprodução e verificação dos resultados de forma independente [33]. Apesar da reprodução científica ser um modelo de investigação bem respeitado em muitas áreas, as barreiras tecnológicas dificultam a transferência da pesquisa de um ambiente de origem para outro de destino. Isto se deve à variedade de diferentes infraestruturas de *hardware* e *software*. Quando um cientista trabalha com *hardware comodity* e utiliza *softwares* básicos como, por exemplo, sistema operacional e compiladores de código aberto e licença de distribuição gratuitos, a reprodução pode ser alcançada por outros pesquisadores [34].

As pesquisas computacionais atuais estão cada vez mais orientadas a dados. Essa característica demanda o uso de ambientes computacionais especializados (*software* e *hardware*) para executar as devidas análises nos conjuntos de dados de uma pesquisa científica. Geralmente, a incapacidade de obter essa infraestrutura especializada impossibilita a reprodução do experimento. Por exemplo, Abadi [35] apresenta cinco

desafios na área de armazenamento e gerenciamento de dados em aplicações de *Big Data* que devem ser considerados em termos de reprodução da infraestrutura de um experimento: (1) infraestrutura de dados alta/rapidamente escalável, (2) gerenciamento de diferentes representações dos dados, (3) processamento e interpretação dos dados ponto a ponto, (4) serviços implantados na nuvem, e (5) o papel das pessoas (agentes) no ciclo de vida do dado.

Na pesquisa reprodutível, o produto final de uma publicação não é apenas o artigo científico, mas engloba todos os objetos de pesquisa necessários para a sua produção. Em geral, uma pesquisa reprodutível é composta pelo conjunto de dados de entrada, *softwares*, bibliotecas, parâmetros de configuração, *workflow* científico, SWfMS, *hardware* e infraestrutura em geral utilizados para a execução de um experimento. O esforço necessário para limpar e documentar os objetos de pesquisa para prepará-los para a reutilização é uma das maiores barreiras para o compartilhamento de uma pesquisa [36, 37]. Portanto, é preciso desenvolver abordagens que reduzam essas barreiras, criando mecanismos que facilitem a captura de detalhes experimentais, bem como, para a comunicação, compartilhamento do ambiente, algoritmos, dados e o raciocínio (em geral na forma de anotações) aos colaboradores e público em geral.

Algumas iniciativas e abordagens de reprodução encorajam o uso de máquinas virtuais para empacotar e distribuir uma pesquisa reprodutível, no entanto, ainda existe o desafio em relação a infraestrutura especializada para reproduzir os experimentos intensivos de dados e de processamento. Além disso, a maioria das ferramentas de *software* não fornecem mecanismos para empacotar uma análise computacional de forma que ela possa ser facilmente compartilhada e reproduzida [38]. Portanto, esta tese de doutorado foi desenvolvida para tratar as questões relacionadas a distribuição de uma pesquisa computacional reprodutível e da infraestrutura necessária para permitir a reprodução do ambiente e resultados.

1.3 Objetivos

O objetivo geral é desenvolver uma abordagem de reprodução de experimentos científicos computacionais. Essa abordagem se baseia em experimentos modelados como *workflows* científicos, tirando proveito da coleta, armazenamento e gerenciamento de dados de metadados e dados de proveniência. As tecnologias de VMs são usadas para preservar as características computacionais, tais como a arquitetura do sistema e os componentes de *hardware* e *software* no qual o *workflow* foi originalmente executado. Essas VMs produzidas podem ser instanciadas e executadas sob demanda nos provedores de computação em nuvem. A computação em nuvem será adotada como um mecanismo para prover a infraestrutura necessária

para reproduzir os experimentos para pemitir atingir de forma eficaz todos os níveis de reprodução. Desta forma, o desafio da reprodução de experimentos executados em ambientes de execução paralela, especialmente aqueles que envolvem um grande volume de dados, fazem uso da elasticidade e computação de alto desempenho das nuvens.

1.3.1 Objetivos Específicos

- Desenvolver um mecanismo de monitoramento da execução para a coleta de metadados e dados de proveniência em nível de sistema operacional e máquinas virtuais sob experimentos computacionais projetados em *workflows* científicos. Este monitoramento auxilia na identificação e obtenção de todos os elementos necessários para a reprodução a posterior reprodução.

- Implementar uma estratégia para preservar o ambiente operacional no qual um experimento científico foi originalmente concebido através da tecnologia de virtualização. A estratégia deve encapsular o ambiente com os componentes de *hardware*, *software*, arquivos de dados e configurações em máquinas e discos virtuais. Essa implementação deve resultar em um pacote de pesquisa reprodutível com todos os elementos encapsulados que permita reproduzir o experimento em sua totalidade.

- Conceber um mecanismo para automatizar a implantação de um experimento científico computacional (pacote de pesquisa reprodutível) apoiado pela abordagem proposta em provedores de nuvem pública. Este mecanismo deve permitir a portabilidade do experimento de um ambiente operacional de origem para um destino. Ele deve adaptar os recursos computacionais originalmente utilizados com os que são ofertados pelos provedores, baseado nos metadados e dados de proveniência obtidos com o monitoramento da execução.

- Desenvolver um serviço *web* para permitir a publicação e compartilhamento dos componentes de uma pesquisa científica reprodutível. Esse serviço deve utilizar as ferramentas de gerenciamento de recursos dos provedores de computação em nuvem e os metadados e dados de proveniência para permitir consultas, referências e recuperação dos experimentos científicos e seus elementos encapsulados em uma VM compartilhada na nuvem.

- Conceber um mecanismo de avaliação da reprodução de um experimento computacional, analisando a qualidade dos resultados gerados a partir da reprodução de diferentes *workflows* científicos previamente consolidados na literatura, bem como características quantitativas, tais como desempenho da infraestrutura reproduzida e integridade dos dados compartilhados.

1.4 Contribuições

Este trabalho traz seis contribuições. A primeira reúne os conceitos e terminologias relacionados com a reprodução experimental nas mais variadas áreas do conhecimento. Embora a literatura seja vasta no tema, a terminologia é muito diversa, confusa e muitas vezes ambígua. O objetivo foi oferecer uma definição conciliatória que sirva como um padrão para a pesquisa reprodutível. O resultado foi uma proposta de estrutura hierárquica na forma de taxonomia para classificar os principais termos e características relacionados com este tema, para ser usada como base para orientar pesquisadores sobre o universo de possibilidades de tecnologias para esse fim.

A segunda contribuição apresenta os requisitos funcionais fundamentais para orientar a construção de uma abordagem de reprodução. Estes requisitos foram obtidos após o levantamento bibliográfico e o atual estado da arte que auxiliaram na identificação de um conjunto de trabalhos que foram peças fundamentais para a definição de uma proposta de arquitetura de referência. Essa arquitetura pode ser usada como uma estrutura base para a construção de abordagens de pesquisa reprodutível.

A terceira contribuição refere-se à criação de um mecanismo de monitoramento e gerenciamento de dados primários (fonte primária/dados de entrada), intermediários, resultados finais, dados derivados, metadados e dados de proveniência. O objetivo é permitir a reconstrução da infraestrutura computacional (*hardware/-software*) de um experimento científico a partir de um ambiente computacional de origem, tais como uma estação de trabalho ou ambiente de *cluster*, para um ambiente de destino, utilizando a infraestrutura e os recursos dos provedores de computação em nuvem. Esta definição permite a portabilidade, publicação, compartilhamento e posterior reprodução de um experimento por terceiros.

A quarta contribuição foi a concepção de um mecanismo de automação do processo de implantação da infraestrutura de computação paralela e distribuída para fins de reprodução de experimentos que usam estratégias de computação de alto desempenho para a execução das suas análises. A estrutura desse tipo de ambiente exige a adoção de recursos especiais, tais como sistemas de arquivos distribuídos e a recuperação de sistemas de computadores homogêneos e heterogêneos. Além disso, foi necessário considerar as restrições de acesso a informações dos recursos em nível de sistemas, no caso de arquiteturas gerenciadas por um instituto e compartilhadas por vários usuários, por exemplo as infraestruturas de *clusters*.

A quinta contribuição foi relativa ao desenvolvimento de uma abordagem de avaliação da reprodução de um experimento científico. Essa abordagem foi concebida a partir dos mecanismos de verificação e a validação dos dados primários, inter-

mediários, resultados finais, metadados e dados de proveniência obtidos através do monitoramento do ambiente de execução em nível de sistemas operacionais. Foram aplicadas as teorias de isomorfismo de grafos e subgrafos para analisar os caminhos de execução e reprodução dos experimentos computacionais a partir dos metadados e dados de proveniência sob os modelos de proveniência OPM e W3C PROV.

A sexta contribuição está relacionada ao potencial de reprodução desenvolvido a partir dos estudos de caso usados nesta tese. É possivel usar os objetos de pesquisa originais e aplicar a abordagem proposta pra reproduzir os resultados aqui obtidos. Consecutivamente, a pesquisa reproduzida pode sofrer uma nova reprodução e os novos resultados podem ser avaliados em relação aos anteriores, para verificar a efetiva reprodução dos novos resultados obtidos.

1.5 Organização do Trabalho

Além deste capítulo, esta tese está organizada em outros seis capítulos. O Capítulo 2 aborda os conceitos relacionados com a repetição e reprodução de experimentos apoiados pela ciência da computação. O Capítulo 3 apresenta um conjunto de trabalhos relacionados com o atual estado da arte de mecanismos de reprodução experimental. O Capítulo 4 apresenta o ReproeScience, a abordagem de reprodução experimental proposta nesta tese, discutindo as etapas envolvidas na concepção, projeto e avaliação. No Capítulo 5 serão discutidos os principais resultados. E ao final, no Capítulo 6, serão apresentadas as conclusões do trabalho e os direcionamentos para trabalhos futuros. O apêndice A traz alguns conceitos e termos complementares relacionados com as tecnologias usadas na construção do mecanismo de reprodução.

Capítulo 2

Reprodução de Experimentos Computacionais

Como a ciência vem se tornando cada vez mais digital, o uso de computadores tem acelerado o seu desenvolvimento, aumentando a eficiência e a eficácia no processo de análise e produção de dados. Entretanto, esse desenvolvimento vem acompanhado de uma produção massiva de dados gerados por um conjunto crescente de simulações computacionais. Ao mesmo tempo aumentam-se os desafios para o desenvolvimento de abordagens que permitam a reprodutibilidade experimental. O principal objetivo é verificar e validar os resultados produzidos por essas simulações.

A reprodução é uma das marcas da ciência [39]. O progresso da ciência depende da efetiva disseminação e reprodução de pesquisas existentes, porém, é necessário ter acesso ao material usado na produção dos resultados para que seja possível validá-los [4]. A ciência avança mais rápido quando é possível construir sobre resultados, quando novas ideias podem ser facilmente medidas contra o estado da arte [40]. Apesar da demanda global, muitos estudos publicados permanecem irreprodutíveis, principalmente devido à falta de disponibilidade do protocolo experimental, dos dados e/ou do código de computador [41].

A publicação de um artigo científico sem os materiais complementares usados para a sua produção não é suficiente para que outros pesquisadores possam verificar os resultados apresentados [36]. Uma publicação científica tem dois objetivos principais, segundo Mesirov [21]: (1) anunciar um resultado obtido por meio de uma pesquisa e (2) convencer os leitores que o resultado está correto. Diante disto, um princípio inerente de uma publicação é que outras pessoas devem ser capazes de validar, reproduzir e construir sobre as alegações publicadas.

A capacidade de reprodução de uma pesquisa científica juntamente com o rigor, transparência e verificação formam um fator decisivo para avaliar a qualidade de um trabalho científico. Uma abordagem transparente e rigorosa trata as questões da reprodução, garantindo o avanço da ciência através de verificações independentes

[42]. Uma pesquisa reprodutível é aquela em que todas as informações relevantes do trabalho, incluindo, mas não limitado ao texto, dados e código, são disponibilizados para que um pesquisador independente possa reproduzir os resultados [43]. A Sociedade Americana de Física (APS)[1] define a ciência, destacando a importância da reprodução, como "o emprego sistemático de coleta de conhecimento sobre o universo e a organização e condensação desse conhecimento em leis e teorias testáveis, sendo que uma das formas de manter a credibilidade e o sucesso da ciência está ancorada na disposição dos cientistas em expor as ideias e resultados dos seus estudos para testes independentes e replicação desses testes por outros cientistas".

Uma descoberta científica é um processo iterativo, onde um estudo publicado gera novos conhecimentos e dados, resultando em novos estudos e ensaios com base nesses resultados [41]. A reprodutibilidade é o meio que permite que o conhecimento publicado torne-se realmente disponível para o público geral [44]. Uma contribuição científica é considerada valiosa se, entre outras coisas, outros pesquisadores forem capazes de reproduzir os seus resultados com sucesso (verificabilidade), para que diante disto, novos trabalhos sejam desenvolvidos e, consequentemente, sejam geradas novas descobertas científicas [33].

Este capítulo apresenta uma definição para os conceitos de reprodução e pesquisa reprodutível com o objetivo de fundamentar o contexto abordado nesta tese. Serão levantados os diversos aspectos e funções relacionadas com a reprodução de uma pesquisa científica na ciência da computação. O principal objetivo é a definição de uma linguagem padronizada para a definição de um fundamento básico para a construção de uma abordagem que permita a automação das tarefas relacionadas com a reprodução científica em soluções de pesquisa que adotam a ciência da computação como ferramenta de gerenciamento de pesquisa científica e análise de resultados computacionais. Este capítulo está organizado em quatro seções. A Seção 2.1 apresenta os principais conceitos e termos relacionados com a reprodução científica. A Seção 2.2 lista as principais vantagens para a adoção e concepção de abordagens de pesquisa reprodutível. A Seção 2.3 discute os principais desafios relacionados com a reprodução. A Seção 2.4 apresenta as principais iniciativas e esforços de conferências e revistas para estimular a adoção de pesquisa reprodutível.

2.1 Reprodução na Ciência da Computação

A ISO/DTS 21748 [45] apresenta os conceitos de repetição e reprodução. A repetição, também definida como replicação, ocorre quando os resultados de testes independentes são obtidos com métodos e itens de testes idênticos, em um mesmo laboratório, pelo mesmo operador, utilizando o mesmo equipamento, com curtos in-

[1] http:// www.aps.org/policy/statements/99_6.cfm

tervalos de tempo. Na reprodução, os resultados do teste são obtidos com o mesmo método, com itens de teste idênticos, porém, em laboratórios diferentes, com operadores e equipamentos diferentes [45]. Fomel e Claerbou [46] também definem duas classes distintas, a repetição e a reprodução. A repetição usa os mesmos processos, dados e demais artefatos de forma equivalente ao experimento original publicado, sem qualquer modificação dos elementos. A reprodução repete o experimento, porém, possibilitando variações no valor ou uso de um ou mais artefatos. A diferença básica entre os dois conceitos está relacionada com as condições experimentais. A repetição é sensível a variações no ambiente experimental, e a reprodução é robusta a essas variações [47].

Freire *et al.* [48] apresentam dois conceitos para definir a reprodução na computação: (1) Um experimento feito por um laboratório L em um tempo t é considerado reprodutível se ele puder ser repetido em um laboratório diferente L' em um tempo posterior t', e (2) um experimento computacional desenvolvido em um tempo t em *hardware*/sistema operacional s utilizando os dados d, é reprodutível se ele puder ser executado no momento t' em um sistema s' com os dados d', que são semelhantes (ou potencialmente os mesmos) a d. A reprodução pode ser verificada por meio de métodos analíticos baseados em medições e métricas sob os resultados gerados pela execução do experimento. Devem ser considerados conceitos baseados em fundamentos matemáticos que usam métrica tais como: polarização, veracidade e precisão dos resultados [45]. A polarização é um valor de referência que analisa a diferença entre os valores esperados e os resultados obtidos para verificar se o resultado está dentro dos padrões de aceitação. A veracidade define o grau de concordância entre o valor médio obtido a partir de um conjunto de resultados dos ensaios e um valor de referência aceito. A precisão é o grau de concordância entre os resultados de teste independentes obtidos em condições previamente estipuladas.

Freire *et al.* [48] propõem três rótulos para descrever os níveis de reprodução de um trabalho científico, baseado em experiências com os esforços de repetição experimental em conferências do Grupo de Interesse Especial em Gerenciamento de Dados (*Special Interest Group on Management of Data* - SIGMOD) da *Association for Computing Machinery* (ACM). Esses níveis auxiliam a identificar o quanto um experimento científico é passível de reprodução, e são definidos da seguinte forma:

- Profundidade/Transparência - *transparency* - determina o quanto de um experimento está disponível ou arquivado. Existem cinco definições para caracterizar esse fator [48]: (1) um conjunto de figuras associado ao artigo, (2) o *script* usado para gerar as figuras contidas no artigo, junto com o conjunto de dados necessários, (3) os dados brutos usados para gerar as figuras contidas no artigo, juntamente com o *script* e os dados derivados usados na geração da figura, (4) o conjunto de experimentos usados para produzir os dados brutos (sistema de

configuração e inicialização, *workflow*, *scripts*, protocolos de medição e etc.), e (5) o sistema de *software* contendo os códigos fonte, documentação, ambiente operacional, ou um arquivo executável que executa o experimento.

- Portabilidade - *portability* - identifica em que ambiente operacional um experimento pode ser reproduzido. Existem três classificações para esse fator: (1) somente no ambiente original, ou seja, não é possível reproduzir o experimento em um ambiente diferente, (2) Pode ser colocado em ambiente similar, com o mesmo sistema operacional, porém, em diferente *hardware*, e (3) em diferentes ambientes, com sistema operacional e *hardware* diferentes do original.

- Cobertura - *coverage* - determina o quanto de um experimento pode ser reproduzido, ou seja, quais processos e artefatos podem ser reutilizados e reexecutados. Este nível apresenta duas classificações: (1) parcial, ou (2) totalmente reproduzido.

A reprodução é definida, em linhas gerais, como um fenômeno previsto de se repetir mesmo quando as condições experimentais variam em um determinado grau. Ela dá a capacidade de realizar mudanças no experimento, para testar a robustez do método mediante variações nos dados e ambiente experimental. A reprodução permite construir novas variações a partir de um método já consolidado. A obtenção de um mesmo resultado experimental mediante a variação das condições do experimento implica em uma robustez do achado original. A repetição é uma classe de reprodução que tem o objetivo de repetir fielmente cada um dos elementos do experimento original. O principal objetivo é verificar se o resultado final do experimento foi fielmente repetido. Trata-se da capacidade de se obter um resultado idêntico quando um experimento é realizado em condições estritamente idênticas.

Um relatório gerado após o Workshop de Reprodução em Ciência da Computação e Matemática organizado pelo ICERM (*Institute for Computational and Experimental Research in Mathematics*[2]) sugeriu uma classificação das pesquisas reprodutíveis baseadas no nível de acesso aos documentos e materiais usados para a construção do experimento, disponibilização dos dados e códigos fonte de programas, em regras e restrições de licenciamento, direitos autorais, direito de cópia e distribuição dos materiais relacionados com a pesquisa. Esse relatório apresenta cinco classes [49]: pesquisa revisável, replicável, confirmável, auditável e reprodutível.

A pesquisa revisável permite que as descrições dos métodos de pesquisa tenham acesso abertos para que os resultados possam ser julgados quanto a sua credibilidade. A pesquisa replicável permite a duplicação dos resultados, por exemplo, por meio da execução de um código fonte e produção dos gráficos mostrados na publicação.

[2]http://icerm.brown.edu/home/index.php

A pesquisa confirmável permite que os resultados possam ser alcançados de forma independente através de uma descrição completa do algoritmo e da metodologia, sem a necessidade de usar o *software* fornecido pelo autor. Na pesquisa auditável os registros dos materiais usados são arquivados para que a pesquisa possa ser defendida posteriormente. Os materiais podem ser privados e podem ser apresentados a um revisor requisitado para uma publicação. A pesquisa reprodutível é abertamente disponível, bem documentada e com o código e dados disponíveis ao público. Ela permite auditoria, replicação e reprodução dos resultados de forma independente e, ainda permite a ampliação e aplicação do método em novos problemas.

A reprodução envolve a reprodução da metodologia, resultados finais e do conjunto de elementos usados para a derivação desses resultados, denominados como objetos de pesquisa. Os objetos de pesquisa são compostos pelos diferentes artefatos usados ou gerados por uma pesquisa [50]: artigos (manuscritos), anotações, conjunto de dados, documentação, infraestrutura de *hardware* e *software*, parâmetros de configuração e assim por diante. Os objetos de pesquisa são como uma abstração usada para a comunicação, compartilhamento e reuso de resultados de pesquisa [50]. Portanto, na computação uma pesquisa reprodutível é aquela cujo produto final é composto pelo artigo, no formato digital ou físico, acompanhado de todos os objetos de pesquisa utilizados para a sua concepção [51].

O compartilhamento dos objetos de pesquisa facilita o desenvolvimento da ciência, não apenas no processo de validação dos resultados, mas também pela possibilidade de exploração de novas hipóteses e combinação com outros métodos. Além disso, para tornar uma pesquisa mais consistente e confiável é necessário criar uma abordagem que torne a reprodução mais conveniente, através de uma combinação de melhores práticas com um conjunto de ferramentas automatizadas [52], provendo acesso aos objetos necessários para sua execução, de uma forma simples e conveniente, com o mínimo esforço e com o poder computacional necessário [23].

Para verificar os resultados computacionais de uma pesquisa, é preciso recriar as condições em que o experimento foi realizado a partir do zero [53]. Esse contexto motiva o desenvolvimento de soluções de pesquisa reprodutível para fornecer os mecanismos necessários para permitir a reprodução de um experimento e das suas conclusões com base nas medidas e métricas estabelecidas para a avaliação dos resultados [54]. Diante dessa necessidade, foi criado o conceito de artigos executáveis (*Executable Paper*). Os artigos executáveis são definidos como um único objeto digital publicado que possui um manuscrito na forma de artigo que descreve um estudo científico e de seus resultados, bem como todo o código necessário para reproduzir os cálculos e visualizar os resultados de dados e programa [39]. O principal objetivo é aumentar a compreensão e a reprodução das publicações eletrônicas, permitindo que os leitores e revisores possam interagir, validar e explorar os experimentos.

2.2 Vantagens dos Experimentos Reprodutíveis

O compartilhamento dos materiais de uma pesquisa reprodutível beneficia a comunidade científica nos mais variados aspectos. Ele incentiva a análise de múltiplas perspectivas sob cada um dos elemento usados, auxilia na identificação de erros nos métodos, materiais e procedimentos, desestimula a fraude, torna-se uma ferramenta útil para a formação de novos pesquisadores e permite o uso eficiente de recursos e financiamento, pois evita a repetição de trabalho [55]. A pesquisa reprodutível estabelece que um experimento relatado e de fato reprodutível mostra os acertos e também os caminhos que não produzem bons resultados, e portanto, torna-se um importante passo para a transferência de conhecimento e tecnologia [56]. Além disso, diminui a quantidade de tempo e esforço demandado para preparar o ambiente operacional para a reprodução dos resultados [53].

A reprodução permite a avaliação de diferentes aspectos dos experimentos, tais como os materiais, métodos e ferramentas empregados. Freire *et al.* [48] listam quatro grandes benefícios: (1) programas reprodutíveis podem ser comparados baseado em termos de desempenho e qualidade dos resultados gerados a partir de conjuntos de dados de entrada em comum, (2) *software* e resultados são documentados permitindo aos usuários entender como usar e modificar o *software*, bem como os conjuntos de dados usados por ele, (3) *software* reprodutível é portável, ele pode ser transferido de um ambiente computacional para ser usado em outro, e (4) os experimentos reprodutíveis são citados, pois permitem que terceiros validem os resultados aumentando o nível de confiança.

A adoção de pesquisa reprodutível pode beneficiar os pesquisadores por meio do aumento da quantidade de citações que referenciam o seu trabalho. A quantidade de citações é uma métrica frequentemente usada por agências no processo de decisão de financiamento para projetos de pesquisa [55] estimulando a produção de trabalhos com essas características.

Cada computação (ensaio experimental) torna-se um teste após o experimento ser concluído, pois, quando o cálculo é repetido no futuro já existe uma definição de valor esperado, um índice de referência. E finalmente, reprodutibilidade requer manutenção, e manutenção requer uma comunidade, pois a falta de manutenção faz com que os experimentos computacionais percam sua reprodutibilidade, consequentemente, perdendo a sua utilidade e credibilidade. O acesso e teste contínuo da comunidade mantém os resultados relevantes reprodutíveis, e o conhecimento associado continuará a aumentar.

2.3 Desafios na Pesquisa Reprodutível

O progresso na ciência é geralmente dificultado pela incapacidade da reprodução e verificação dos resultados de forma independente [33]. Têm aumentado as discussões sobre uma crise de credibilidade na computação, pois, atualmente é impossível verificar a maioria dos resultados computacionais apresentados e submetidos em conferências [5, 44, 57]. Mesmo a computação se posicionando ao lado da ciência teórica e experimental em grau de importância, ela ainda não atingiu o mesmo grau de maturidade em termos de reprodução[37]. Uma pesquisa científica é considerada de boa procedência quando ela é documentada com detalhes suficientes para permitir a sua reprodução [22].

Apesar da reprodução científica ser um modelo de investigação bem respeitado, as barreiras tecnológicas dificultam a transferência da pesquisa de um ambiente de origem para outro de destino devido à variedade de infraestruturas de *hardware* e *software*. Quando um cientista trabalha com *hardware comodity*, e utiliza *softwares* básicos, como sistema operacional e compiladores de código aberto e licença de distribuição gratuitos, a reprodução pode ser alcançada por outros pesquisadores [34]. Entretanto, essas plataformas vão exigir que os usuários obtenham, configurem e usem esses conjuntos de elementos disponibilizados. Existe, ainda, a necessidade de padronizar os mecanismos de armazenamento, consulta e comparação dos experimentos para que eles possam ser facilmente recuperados e, consequentemente, mais úteis para futuros pesquisadores [48].

Diversos trabalhos publicam os métodos aplicados de forma vaga apresentando algoritmos que muitas vezes possuem poucos detalhes de implementação ou são tão complicados que dificilmente será possível implementá-los corretamente [37]. Além disso, grande parte dos resultados irreprodutíveis são consequência de falhas estatísticas, incluindo erros no projeto, análise e interpretação dos dados [58].

Vitek e Kalibera [59] destacam que a avaliação de um método na computação requer a execução de *benchmarks* em diferentes plataformas de *hardware* e *software*. Além disso, para fazer uma comparação com o estado da arte é necessário implementar as abordagens concorrentes. Keiding [60] destaca que a reprodutibilidade por si só não é a forma mais ampla de verificação dos resultados. É necessário verificar a quantidade e a qualidade dos dados usados nos experimentos para identificar o quanto eles representam o contexto. Desta forma, os dados, parâmetros de configuração, ferramentas de *hardware*, *software* e infraestruturas usados na pesquisa devem passar por uma análise e refinamento por outros cientistas.

Pesquisas orientadas a dados possuem desafios relacionados com a área de armazenamento e gerenciamento de dados. Abadi *et al.* [35] descreve os cinco principais desafios na área de dados: (1) infraestrutura de dados alta/rapidamente escalável,

(2) gerenciamento de diferentes representações dos dados, (3) processamento e interpretação dos dados ponto a ponto, (4) serviços implantados na nuvem, e (5) o papel das pessoas (agentes) no ciclo de vida do dado. Estes desafios, em termos gerais, abrangem as características das aplicações de *Big Data*, que reúnem características de volume, velocidade e variedade, assim como, as característica do ambiente e o envolvimento das pessoas em aplicações contendo essas características [35].

Atualmente, têm sido apresentadas diversas soluções de *software* inovadoras para o problema da reprodução [38]. Grande parte delas são bem projetadas tecnicamente, passando por um processo de construção bem definido, aplicando-se técnicas de análise, representação, padronização e compartilhamento de dados e experimentos científicos. Porém, grande parte fracassa devido a diversos fatores [38], tais como: proteção de formatos de arquivos proprietários, falta de recompensa pela academia e agências de financiamento, dificuldade na concepção de soluções genéricas e a exigência da rápida publicação dos resultados se opõem ao processo de desenvolvimento baseado em padrões de projeto de *software*.

Fomel [61] apresenta um conjunto de lições sobre reprodução obtidas no desenvolvimento do projeto Madagascar [62]. Segundo Fomel, reprodução não é o objetivo, o importante na realidade é tornar a pesquisa mais fácil de ser verificada e aplicada. A replicação em cálculos com números de ponto flutuante e erros de arredondamento é difícil de ser obtida. Essas lições são reforçados por Baker [63] e Loscalzo [58]. Para Baker a impossibilidade de replicação das conclusões iniciais não determina que o resultado original está errado, ele pode estar certo, porém, é difícil de reproduzir. Já Loscalzo destaca que mesmo que uma observação reprodutível seja considerada verdadeira e uma irreprodutível seja falsa, não se pode misturar os conceitos de verdade e reprodução.

O investimento em testes diminuem os problemas relacionados ao mau funcionamento do código nas mãos de outros usuários. A adoção de um mecanismo de gerenciamento de código fonte permite que o cientista preserve a versão exata do código que produziu os resultados publicados [64]. Em seguida, a padronização de ferramentas e plataformas usadas na execução de um experimento, em termos de bibliotecas e ambientes computacionais, reduzem a complexidade de implantação do ambiente para a reprodução. Além disso, é preciso padronizar ferramentas e plataformas usadas na execução de um experimento.

Goble [22] lista seis fatores chave para o desenvolvimento de soluções de reprodução: (1) as informações sobre o experimento podem estar em diferentes repositórios, sendo necessário desenvolver uma estratégia de identificação única para referenciar autores e os objetos de pesquisa, (2) a existência de dependências, pois é necessário identificar e manter os artefatos usados no experimento, ou pelo menos a referência a eles, (3) a gestão de modificações trata a prevenção, detecção e reparo

em dados e no próprio método quando ocorrem modificações, pois um dos objetivos da reprodução é a capacidade de modificação para testar novos parâmetros e dados, (4) a documentação rica, que minimiza o impacto de implantação para o usuário, (5) a descrição compreensiva e explícita o suficiente para auxiliar na definição dos elementos a serem reproduzidos, (6) os recursos como a *web*, anotações e artigos executáveis para facilitar esse processo.

A publicação de *softwares* reprodutíveis, segundo Donoho *et al.* [65], exige o tratamento das seguintes questões: (a) suporte técnico aos usuários, por causa de problemas muitas vezes ocasionados por mudanças de versões de *kernel* de sistema operacional ou de *softwares* e bibliotecas cujo *software* é dependente, (b) manutenção do *software*, tanto para a correção de erros identificados no uso diário, quanto para a adaptação a novas arquiteturas ou implantação de melhorias, e (c) manutenção de uma página *web* como um repositório de informações e novas versões do *software*. Além disso, o *software* deve estar disponível para o público durante anos.

O esforço necessário para limpar e documentar os códigos e dados para prepará-los para a reutilização é uma das maiores barreiras para o compartilhamento dos objetos de pesquisa [36, 37]. É necessário desenvolver abordagens que reduzam essas barreiras, criando mecanismos que facilitem a captura de detalhes experimentais, bem como a comunicação e compartilhamento do ambiente, algoritmos, dados e o raciocínio aos colaboradores e público em geral. Além disso, não existe um benefício claro e direto que motive os pesquisadores a demandar esforços na produção deste material complementar.

Um estudo científico deve ser documentado de tal forma que outro pesquisador possa seguir os mesmos passos e obter os mesmos resultados [39]. A documentação é um elemento indispensável para auxiliar terceiros a entender como implantar e usar o *software*. A documentação da pesquisa deve ter um diagrama de blocos, com as etapas e sequências de atividades para a obtenção dos resultados, e um pseudocódigo bem detalhado para que outros cientistas possam reimplementar algoritmos sem incertezas. Além disso, uma lista dos parâmetros de configuração e os respectivos valores usados para o experimento devem ser claramente indicados.

A publicação e compartilhamento dos objetos de pesquisas científicas deve considerar desafios quanto a questões de licença e direitos de autorais. Em geral, a maioria dos objetos de pesquisa científica são inéditos a comunidade acadêmica. A lei de direito autoral é uma estrutura legal inadequada para trabalhos científicos, pois as normas científicas guiam para a reprodução e construção de novas soluções sob a pesquisa de outros, contrariando o padrão da lei de direitos autorais [66]. Portanto, é preciso vincular licenças específicas segundo as características de cada tipo de objeto de pesquisa permitindo a reprodutibilidade.

2.4 Esforços para a Reprodução

Esforços para a garantia da repetição e reprodução experimental têm sido executados em iniciativas como o SIGMOD *Experimental Repeatability* [67], que tem se dedicado a definir diretrizes para a repetição e reprodução. O principal objetivo é garantir que os artigos científicos submetidos na conferência sejam confiáveis e referenciáveis para pesquisas futuras. O SIGMOD apresenta como melhores práticas o uso de VMs para o encapsulamento do ambiente de trabalho e motores de *workflow* científico para automatizar a preparação dos experimentos. Tais práticas e tecnologias minimizam os esforços para a preparação e documentação de um experimento.

Os trabalhos submetidos ao SIGMOD são rotulados como reprodutíveis ou compartilhados [67]. Os reprodutíveis são submetidos junto com o protocolo e recursos necessários para reproduzir os resultados. Os recursos englobam: (1) o protótipo do sistema, com o código fonte, arquivos de configuração e ambiente operacional, (2) o conjunto de experimentos, contendo os sistemas de configuração e inicialização, *scripts* e protocolos de avaliação usados para produzir os resultados, e (3) os *scripts* necessários para transformar os dados em equações, gráficos e tabelas apresentados no artigo. Os compartilhados devem apresentar um *link* para uma URL com os recursos necessários para realizar os testes e a validação dos resultados apresentados.

As diretrizes para a repetição e reprodução do SIGMOD concentram-se na forma como os dados dos experimentos são obtidos e gerados. Em geral, esses dados são classificados em dados primários e derivados [67]. Os primários são dados brutos sob os quais as conclusões são obtidas. Eles são obtidos através da reexecução de um *software* ou por um *hardware* especial. Nesse caso é necessário fornecer um protocolo detalhado contendo os procedimentos para a configuração do sistema, experimentos e as devidas métricas de medição. Os dados derivados são resultados dos processos de medição experimental. Eles são usados para a definição de elementos nas publicações científicas, tais como gráficos, tabelas e imagens. Nesse caso, os autores podem fornecer os *scripts* ou planilhas usadas para calcular as derivações.

Na área da biomedicina e biotecnologia o tema reprodução se tornou tão preocupante que foi criada a Iniciativa de Reprodução [68]. Ela tem o objetivo de identificar e premiar pesquisas reprodutíveis de alta qualidade por meio da validação independente dos resultados. Os experimentos nesta iniciativa são verificados e validados por laboratórios especializados na reprodução dos resultados de forma confidencial e em sigilo [68]. Os trabalhos validados pela iniciativa recebem uma certificação de validação independente e pode ter seus resultados publicados na coleção de reprodução PLOS, com os dados compartilhados no *figShare* e o impacto rastreado na coleção de reprodução do Mendely. A iniciativa foi criada através de uma parceria entre a *Exchance Science, PLOS, FigShare e Mendeley*, definidos a seguir [68]:

- O *Science Exchange* [3] é um tipo de mercado para a colaboração científica, onde os pesquisadores encomendam a validação dos experimentos para os melhores laboratórios, para melhorar a qualidade e eficiência da investigação científica.

- O PLOS [4] é um projeto sem fins lucrativos de acesso aberto a publicação científica. Ele visa criar uma biblioteca de revistas com a literatura científica sob uma licença de conteúdo aberto.

- O figshare [5] é um repositório de dados onde os usuários podem disponibilizar todos os seus resultados de pesquisa de forma citável e compartilhável.

- O Mendeley [6] é um gerenciador de referência e rede social acadêmica livre que permite aos usuários criarem a sua própria biblioteca, permitindo citar, ler e anotar trabalhos a partir de qualquer dispositivo.

A iniciativa de reprodução é uma maneira de plataformas *on-line* facilitarem a comprovação rápida e independente de resultados publicados. Porém, ainda são necessários importantes progressos para a melhoria nos processos de reprodução [68].

Outra iniciativa para a repetição e reprodução foi realizada no ano de 2011 no Grande Desafio de Artigos Executáveis (*The Executable Paper Grand Challenge* [11]), organizado pela Elsevier. O evento foi realizado na forma de uma competição cujo objetivo foi selecionar as melhores abordagens de concepção e gerenciamento de artigos executáveis, melhorando a forma como a informação científica é comunicada e utilizada [11]. A chamada para apresentação de trabalhos listou um conjunto de diretrizes que atendem aos requisitos básicos para a concepção de artigos científicos executáveis, que são apresentadas na tabela 2.1. O desafio contou com a submissão de setenta artigos, dos quais dez foram selecionados como finalistas.

A revista *Nature* (*Nature Publishing Group*) [7] disponibiliza um tutorial com os requisitos básicos para que os autores submetam e disponibilizem os dados, materiais e métodos usados na produção de uma pesquisa científica [69]. O periódico segue o princípio de que outros pesquisadores devem ser capazes de replicar e construir novos trabalhos a partir das alegações transcritas em uma publicação submetida para a revista. É enfatizado que os autores são obrigados a construir materiais, códigos, dados e procedimentos para que os leitores consigam reproduzir o experimento. As orientações para a disponibilização desses materiais são listados a seguir [69]:

- Dados: devem ser apresentados aos editores e revisores na submissão do artigo, para que os resultados possam ser avaliados. Para grandes conjuntos de dados,

[3] https://www.scienceexchange.com/about
[4] http://www.plos.org/
[5] http://figshare.com/
[6] http://mendeley.com/
[7] http://www.nature.com/

Tabela 2.1: Diretrizes para submissão de propostas para o Desafio de Artigos Executáveis da Elsevier [11].

Diretriz	**Descrição**
Executabilidade	Como conceber equações, gráficos e tabelas para que revisores e leitores possam checar e explorar o espaço de resultados. Como tornar esses elementos disponíveis para permitir o experimento ser repetido e manipulado.
Compatibilidade a curto e longo prazo	Como desenvolver um modelo para arquivos executáveis que seja compatível com as arquiteturas e sistemas operacionais dos usuários. Além disso, como adaptar este modelo para sistemas no futuro.
Validação	Como validar dados e códigos e diminuir a carga de trabalho do revisor.
Licenciamento e Direito de Cópia	Os dados dessa natureza devem estar sempre disponíveis para os pesquisadores. Como encorajar este princípio mantendo a patente e a proteção ao direito intelectual do autor.
Sistema	Como transmitir trabalhos realizados em computadores de grande porte que são disponibilizados a uma pequena parcela da comunidade.
Tamanho	Como gerenciar grandes conjuntos de dados.
Proveniência	Como apoiar o registro e acompanhamento das ações tomadas sob o artigo.

é orientada a utilização de repositórios públicos, ou em alguns casos, os dados podem ser fornecidos diretamente para a revista. Os autores são incentivados a publicar informações sobre os dados (metadados e dados de proveniência) para aumentar a sua transparência e o valor para futura reutilização.

- Materiais: os autores são obrigados a disponibilizar os materiais para terceiros. Se os materiais usados forem distribuídos por uma empresa com fins lucrativos, deve ser informado no artigo. No caso de novos compostos químicos (biologia, biomedicina, bioquímica e etc) deve ser fornecida a estrutura química, síntese e caracterização dos compostos para que seja possível a sua reprodução.

- Código de computador: os autores devem disponibilizar, mediante pedido, qualquer código de computador usado para gerar resultados relatados. Questões legais são consideradas e avaliadas pelos editores que reservam o direito de recusar o artigo caso o código não esteja disponível. Após a publicação, são consideradas as melhores práticas para liberar o código de uma forma que permita que os leitores consigam repetir os resultados publicados.

- Protocolos experimentais: os autores são orientados a compartilhar seus protocolos experimentais, utilizando a ferramenta *Protocol Exchange* [8], que trata-se de um recurso gratuito mantido pela *Nature Publishing Group*.

As políticas de publicação da revista *Science* definem que os autores devem disponibilizar todos os dados e materiais necessários para a compreensão, avaliação e ampliação das conclusões do artigo, inclusive os códigos de programas [70]. Os autores devem informar as políticas de restrições a qualquer um desses materiais, e

[8]http://www.nature.com/protocolexchange/

os casos que envolvem grandes conjuntos de dados devem ser incluídos como materiais complementares do artigo hospedados nos servidores da revista *Science* ou hospedado em *site* institucional [70].

No editorial do periódico *Biostatistics* apresentado por Peng [71] são consideradas três classes de submissões de artigos: Dados, Código e Reprodutível. A classe dados refere-se à disponibilização dos dados que provêm os principais resultados, sendo os autores os responsáveis pela permissão de publicação e disseminação desses dados. A classe código agrega qualquer código de computador usado para gerar os resultados publicados. E a classe reprodutível incorpora artigos que permitem executar o código nos dados fornecidos e produzir resultados semelhantes aos que os autores afirmaram ser reprodutíveis. Deve-se atentar ao tipo de resultado fornecido pela pesquisa, em termos de acurácia e precisão.

2.5 Mapa Conceitual

A quantidade de trabalhos e abordagens evidenciam como a reprodução científica tem emergido como um conceito fundamental que já está sendo adotado em muitos domínios científicos como um padrão. Diversas aplicações, plataformas, infraestruturas, tecnologias e padrões tem sido proposto. Contudo, os conceitos envolvidos com a reprodução científica computacional não estão formalizados. Considerando o alto interesse na ciência reprodutível e na dificuldade de encontrar definições organizada de conceito associados a este paradigma será apresentado um mapa conceitual baseado nas iniciativas e abordagens de reprodução para a *e-Science*.

O mapa conceitual organiza os conceitos relacionados com domínio de reprodução científica computacional. Ele fornece uma melhor compreensão do domínio e ajuda na comparação de diferentes abordagens. Ao consultar esse mapa conceitual, um cientista pode considerar apenas as características que atendam a sua necessidade, e dependendo do experimento científico, essas necessidades variam. O mapa conceitual considera uma visão ampla da ciência reprodutível e tem como objetivo explorar os seus principais aspectos. Ao usá-lo como um vocabulário comum é possível facilitar a identificação de características comuns das abordagens existentes facilitando na escolha da mais adequada.

São descritas 7 sub categorias principais apresentadas na Figura 2.1 que compõem o modelo mais geral, classificando as características da pesquisa reprodutível em termos de apresentação, agentes, avaliação, infraestrutura, licenciamento, metodologia de reprodução e tipos de dados. A sub categoria infraestrutura é composta por três elementos, formadas pelos parâmetros de configuração, *hardware* e *software*. Cada um desses elementos serão discutidos em detalhes.

Atores. Uma pesquisa reprodutível pode sofrer a ação de quatro atores segundo

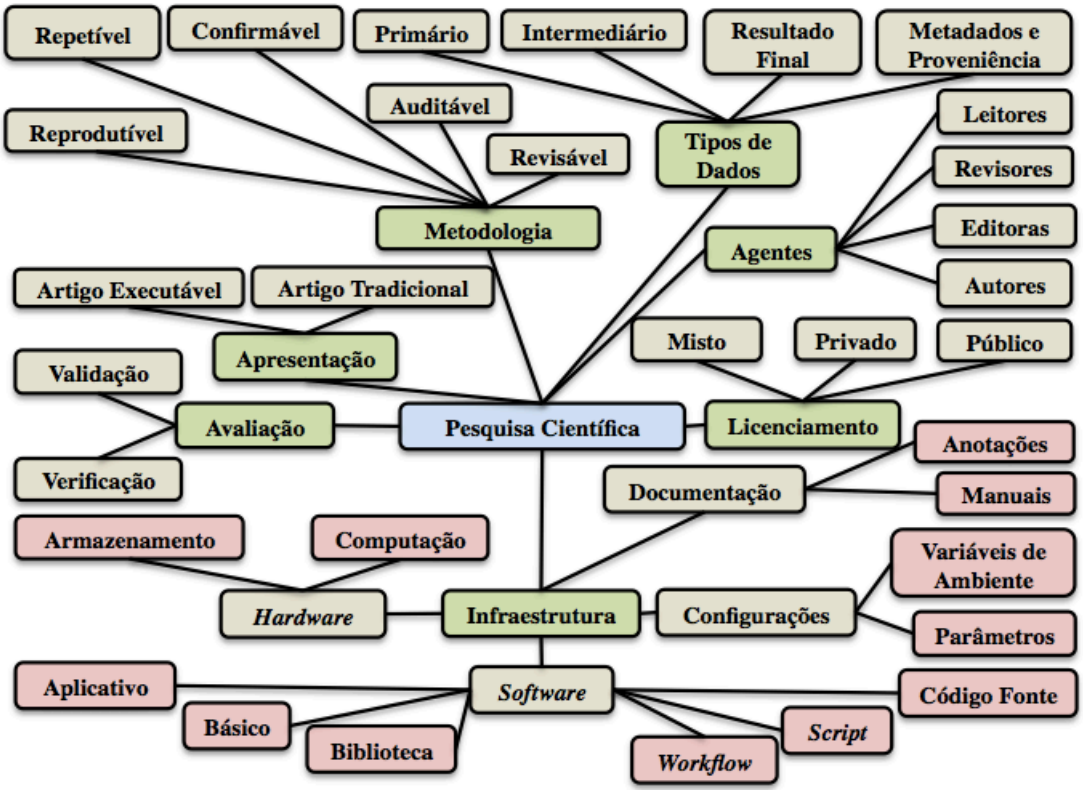


Figura 2.1: Proposta de mapa conceitual para a pesquisa científica reprodutível.

proposto por Koop *et al.* [31]: (1) o autor que projeta e executa os experimentos para gerar os resultados usados no artigo científico, (2) as editoras que recebem e publicam os artigos para que o público possa ter acesso ao conteúdo, (3) os revisores que avaliam o conteúdo e os demais materiais, verificando os resultados e validando a metodologia, e (4) os leitores que acessam o artigo para ter acesso ao seu conteúdo para analisar os resultados, reexecutar o experimento e reutilizar os elementos da pesquisa, por exemplo, para a concepção de novos estudos. O autor de uma pesquisa reprodutível desempenha as seguintes funções: (1) projeto e execução dos experimentos, (2) geração e visualizações dos resultados, (3) escrita do texto e conexão do texto com os resultados e (4) consulta a literatura existente e aos resultados.

Os revisores executam: (1) navegação pelas publicações, experimentos e dados dos experimentos, (2) verificação da conexão das referências apresentadas no texto com os objetos de pesquisa armazenados na base de dados e arquivos, (3) reexecução de experimentos publicados utilizando novos dados e variados parâmetros, (4) adaptação dos experimentos para a reexecução em novos ambientes e (5) consulta a literatura existente e aos resultados. Os leitores compartilham as mesmas funções que os revisores, porém, fazem a reutilização dos resultados obtidos com a pesquisa. As editoras executam as seguintes funções: (1) hospedagem dos artigo, experimentos e dados na sua infraestrutura, (2) fornecimento de opções de consulta a objetos de pesquisa, composição de interface para visualização dos artigos e objetos de pes-

quisa, (3) oferta de formas para a execução de computações sob os experimentos, e (4) adaptação dos experimentos para que possam executar em novos ambientes e infraestruturas computacionais. Este conjunto de funcionalidades podem ser implementadas e fornecidas na forma de métodos para que terceiros possam produzir e consumir as informações e objetos de pesquisa.

Metodologia. Uma pesquisa oferece cinco modalidades de reprodução de um experimento: (1) a repetição define que o resultado de uma pesquisa pode ser reproduzido apenas na infraestrutura que eles foram originalmente obtidos, (2) a reprodução apresenta uma metodologia mais robusta que permite reproduzir os resultados em um ambiente operacional diferente daquele onde ele foi originalmente obtido, (3) a confirmação permite o acesso aos objetos utilizados na execução do experimento, entretanto, não oferece suporte para a reexecução do experimento, (4) a revisão permite que resultados possam ser alcançados de forma independente através de uma descrição completa do algoritmo e da metodologia, sem a necessidade de usar o *software* fornecido pelo autor, e (5) a pesquisa auditável que possui os registros dos materiais usados na pesquisa, porém, são apresentados a um revisor quando requisitado.

Licenciamento. A licença para acesso ou uso dos objetos de pesquisa pode ser definida de três formas distintas: pública, privada ou mista. Quando uma pesquisa é definida pública, o artigo juntamente com os objetos de pesquisa são amplamente acessíveis ao público em geral, ou seja, qualquer pessoa pode obter os objetos usados na pesquisa e reproduzir os resultados e metodologia. Quando uma pesquisa reprodutível é definida como privada, o artigo científico e os objetos de pesquisa tem o acesso controlado a um público específico, mediante uma regra de controle de acesso e distribuição dos objetos de pesquisa. A pesquisa de licença mista possui políticas de concessão e restrição de acesso para cada um dos objetos de pesquisa que pode ser pública ou privada, segundo diversas formas de licenciamento atualmente adotadas.

Os artigos, tabelas, gráficos, equações e demais mídias podem ser associados a licenças *Creative Commons* (CC) que permite o uso das obras desde que seja especificada a atribuição do proprietário [72] [73]. No caso do código, deve ser analisada a distribuição do fonte e binário. As licenças GNU GPL (*General Public License*) incentivam a publicação do *software* juntamente com o código fonte [74], seguindo os seguintes componentes [9]: (1) todo *software* sujeito a essa licença deverá ter o seu código fonte liberado, e (2) uma vez que a licença está ligada ao código, ela também será atribuída a qualquer instituição que utilize o código original. Além disso, pode ser adotada a licença LGPL (*Lesser General Public License*), para códigos de bibliotecas permitindo o uso em pacotes de *softwares* proprietários desde que os produtos derivados informem a existência dessas licenças (GPL e LGPL) [66].

[9] http://www.gnu.org/

A licença BSD (*Berkeley Software Distribution*) Modificada permite a utilização e cópia de códigos fonte modificados e não modificados desde que a licença venha acompanhada do nome dos autores, e devem ser acompanhadas da denominação: Copyright (c) <ANO>, <PROPRIETÁRIO> All rights reserved. Nesse caso, a redistribuição e uso do código fonte e arquivo binário, com e sem modificações pode ser feito quando [66]: (1) redistribui coes mantenham o aviso de direitos autorais, (2) redistribuição dos binários deve manter o aviso dos direitos autorais com a exclusão da responsabilidade sob a documentação e demais materiais derivados, e (3) os nomes da organização e colaboradores podem ser usados para endossar ou promover produtos derivados desde que se tenha uma autorização por escrito.

Existe ainda a licença MIT que é bastante similar a BSB, excluindo a cláusula de proibição do endosso. A licença Apache 2.0 permite o exercício do direito de patente para os trabalhos derivados, permitindo o uso do código em trabalhos derivados desde que sejam assinalados os arquivos que foram modificados. Devem ser informados todos os avisos de direitos autorais, marcas registradas e patentes [66]. Já o caso dos dados, os fatos descritos por um determinado dado são considerados componentes significantes e são de domínio público conforme o Protocolo de Dados Públicos Aberto para Ciência (*Science Commons Open Data Protocol*)[10]. Entretanto, segundo Stodden [66], os produtos originais relacionados com esses fatos possui direitos autorais, ou seja, os fatos descritos em um trabalho são de domínio público, mas a seleção e o arranjo desses dados possuem direitos autorais.

Apresentação. Um artigo científico é o meio para a apresentação dos objetos de pesquisa usados para a produção de um resultado. Atualmente, essa apresentação pode ser feita de uma forma tradicional ou através de artigo executável. Os artigos tradicionais têm o conteúdo estático, permitindo a inclusão de URLs para acesso aos objetos de pesquisa relacionados a um resultado publicado. Os artigos executáveis possuem conteúdo dinâmico, permitindo que os usuários possam informar valores e parâmetros para testar a metodologia e verificar o resultado. Eles pode conter fragmentos de código ou mecanismos que acionam ações para a reexecução da pesquisa.

Avaliação. Uma pesquisa reprodutível deve prover mecanismos para validação e verificação da metodologia e dos resultados experimentais. A verificação avalia se os resultados gerados por uma pesquisa estão de acordo com os experimentos ou observações do fenômeno que está sendo estudado. Por exemplo, Gavish & Donoho [75] apresentam três conceitos para a construção de um método de verificação: (1) Resultado computacional verificável (VCR) com o resultado computacional juntamente com os metadados descrevendo as computações para a geração do resultado, (2) os repositórios de VCR representando um provedor de serviços web para arquivar

[10]http://sciencecommons.org/projects/publishing/open-access-data-protocol/

os VCRs para compartilhamento, e (3) identificador de resultado verificável (VRI) na forma de uma URL que identifica universalmente e permanentemente um repositório que apresenta os VCRs específicos. O objetivo é criar uma assinatura digital criptografada que codifica as informações sobre as condições sob a qual os resultados foram gerados por meio de uma marca d'agua [75]. A validação avalia se o método proposto pela pesquisa resolve corretamente o que ela foi desenvolvida para resolver. A validação pode ser obtida por meio de registros de *log* ou proveniência. Os mecanismos de validação podem ser baseado em trabalhos como o de Missier et al. [76], que analisa a reprodutibilidade baseado em dados de proveniência, analisando as trilhas de execução de um workflow.

Tipos de dados. Os dados de uma pesquisa reprodutível são classificados como: primários, intermediários, resultado final e metadados/dados de proveniência. Os dados primários são usados como dados de entrada dos experimentos. Em geral, são dados obtidos a partir de medições ou monitoramento do ambiente em que um instrumento ou sensor estão implantados. Os dados intermediários, também chamados de derivados, são produzidos durante a execução, sendo resultados da aplicação de algoritmos e técnicas de análises para a extração de informações a partir de dados de entrada. Eles são submetidos a um arranjo ou estrutura para a produção dos resultados finais. Os resultados finais representam o produto final da execução de um experimento (metodologia).

Os metadados e dados de proveniência são, respectivamente, as informações complementares e as informações do histório de derivação das três classes de dados anteriores. Os metadados são os dados descritores que atribuem um significado ao dado, ou seja, são dados sobre dados. Uma das definições mais abrangentes, de Greenberg [77], descreve que metadados são dados que descrevem a estrutura de um objeto e as funções associadas a este objeto. A proveniência é definida como a origem ou histórico de derivação de algum objeto a partir da sua fonte original [78]. Essas informações podem ser usadas para formar avaliações de qualidade, confiabilidade ou confiança de um objeto [79], descrevendo as etapas nas quais os dados foram derivados, e adicionando um valor significante aos dados [80].

Infraestrutura. São todos os recursos de computação usados durante o processo de planejamento, projeto, construção e execução de uma pesquisa reprodutível. A infraestrutura é composta de quatro classes de recursos: documentação, informações de configuração, equipamentos de hardware e software.

Documentação. É composta por todo material digital usado para auxiliar terceiros a implantarem os objetos de pesquisa necessários para reproduzir uma pesquisa e para orientar o entendimento do raciocínio dos pesquisadores empregado para a concepção de cada elemento e da pesquisa como um todo. Guo & Seltzer [81] enfatizam a importância de incluir anotações em produtos de dados usados

e gerados na execução de um experimento, para permitir a recuperação exata de arquivos citados em uma anotação. A documentação é formada, em geral, pelo manual de instruções, pelo conjunto de anotações do autor da pesquisa e demais materiais de apoio.

Configuração. As configurações são o conjunto de dados e informações operacionais responsáveis por tornar o ambiente computacional operacional. O objetivo é que conjunto de *softwares* possa ser implantado e executado sob um conjunto de equipamentos de *hardwares* necessários para a reprodução da pesquisa. As configurações são formadas pelas variáveis de ambiente, que devem ser informadas no(s) sistema(s) operacional(is) onde os objetos de pesquisa serão implantados e os parâmetros de configuração dos *softwares* que serão executados na pesquisa.

Hardware. São todos os equipamentos usados para o processamento das computações de uma pesquisa científica. Na *e-Science* destacam-se os equipamentos que provêm alto poder de computação e grande espaço de armazenamento. O *hardware* é representado pelas infraestruturas de computação, redes de computadores, armazenamento e a organização desse conjunto de elementos, ou seja, a arquitetura do ambiente. A computação representa o par de recursos processador e memória principal. A rede de computadores provém informações sobre a organização lógica e física, tais como, a política de endereçamento IP (Internet Protocol) e latência do fluxo de informações. O armazenamento representa os mecanismos para persistência de dados em uma estrutura de memória secundária (armazenamento de massa). A arquitetura organiza os elementos anteriores para extrair um desempenho e armazenamento mais otimizado, levando em conta características como o uso de computadores físicos ou ambiente virtualizado.

Software. São os elementos que executam as instruções e algoritmos de uma pesquisa. São classificados em seis grupos diferentes: básico, aplicativo, bibliotecas, código fonte, *script* e *workflow*. Os *softwares* básicos são formados pelos sistemas operacionais onde os *softwares* aplicativos são executados, *softwares* de virtualização, também denominados *hypervisors*, e compiladores, responsáveis por gerar um executável de um código fonte fornecido pelo pesquisador. Os *softwares* aplicativos são os programas (arquivos executáveis) usados para a execução de uma atividades específica. Já as bibliotecas são formadas por um conjunto de subprogramas com códigos e dados auxiliares que provém alguns serviços aos *softwares* básicos e aplicativos. As bibliotecas são invocadas durante a compilação e execução, criando-se uma dependência do programa com ela.

Os códigos fonte são os arquivos escritos em uma linguagem de programação prontos para serem compilados e utilizados para a execução de uma atividade. Neste sentido, é importante observar que a reprodução de um determinado resultado por meio do seu código fonte pode ser afetado pelo compilador usado no processo de

compilação do código, portanto, é necessário informar sob qual compilador o código fonte gerou os resultados de uma pesquisa e quais foram os parâmetros aplicados. Um *script* é um arquivo com um conjunto de instruções em código usado pelo sistema operacional para o controle de programas. Um *workflow* é uma abstração que permite a composição de um conjunto de programas e *scripts* na forma de fluxo de atividades, com o objetivo de se atingir um resultado desejado [82]. É executado por um programa aplicativo denominado motor de *workflow* (SWfMS) usado como uma ferramenta especial para automatizar as atividades envolvidas em experimentos da *e-Science*.

O armazenamento do *software* deve levar em consideração quatro propriedades [34]: produto, versão, variantes e instâncias. O produto é a descrição do *software*, ou seja, a forma como ele é conhecido. Os produtos de *software* possuem versões que expressam um conjunto bem definido de funções e comportamento do produto relacionado a esta versão. As variantes de um *software* identificam os ambientes e características operacionais que são acomodados pelo *software*, tais como: sistema operacional, linguagem de programação, bibliotecas etc. A instância é um exemplo do produto, ou seja, a sua instalação que pode ser encontrada em uma máquina em particular.

2.6 Considerações Finais

O contexto da reprodução computacional apresentado neste capítulo levanta algumas características e propriedades que podem orientar no desenvolvimento de soluções e tecnologias para fins de reprodução. São diversos aspectos que ajudam na disseminação do conhecimento adquirido com uma pesquisa científica que precisam de métodos eficazes de comunicação e compartilhamento com a comunidade científica. Diversos trabalhos citados neste capítulo apontam a necessidade de uma documentação mais precisa e o compartilhamento de todos os elementos usados na produção dos resultados. Foram destacados outros fatores como questões relacionadas a licenças e direitos autorais e mecanismos de verificação dos resultados.

Todas as definições, desafios e iniciativas apresentadas neste capítulo apontaram para a necessidade de desenvolver práticas e tecnologias que minimizam os esforços para a preparação e documentação de um experimento científico, de forma que ele possa ser disseminado e, consequentemente, passível de reprodução por terceiros. Além disso, são propostas diretrizes que auxiliam os pesquisadores a desenvolver abordagens que atendam a requisitos explícitos de reprodução científica. Este capítulo apresentou ainda uma compilação dos conceitos relacionados à reprodução na ciência, que serão utilizados para orientar a comparação das abordagens existentes para a investigação reprodutível. Esta compilação evoluiu para um mapa

conceitual, que dirigiu para o desenvolvimento da abordagem de reprodutibilidade proposta nesta tese.

Capítulo 3

Trabalhos Relacionados

Este capítulo apresenta uma visão geral do conjunto de trabalhos usados para o desenvolvimento e comparação da solução proposta. Este conjunto de trabalhos aborda os 4 principais conceitos empregados na abordagem, e são organizados e discutidos em 4 seções. As abordagens de reprodução científicas existentes foram classificadas neste capítulo em 3 categorias: reprodução em nível de sistemas, em artigos executáveis, e reprodução em nuvem computacional.

A Seção 3.1 concentra-se nas abordagens de coleta de proveniência de experimentos em nível de sistemas. A Seção 3.2 traz soluções que tratam os artigos científicos executáveis e a infraestrutura de reprodutibilidade experimental necessária para composição desses artigos. A Seção 3.3 apresenta alguns abordagens e propostas para a reprodutibilidade científica através de recursos de provedores de computação em nuvem. Ao final, na Seção 3.5 serão discutidas a utilização desse conjunto de conceitos e sua relação com na abordagem proposta nesta tese.

3.1 Reprodução em Nível de Sistemas

A coleta de dados para a reprodução pode ser feita em diferentes camadas e níveis de abstração, desde um nível mais abstrato, tal como uma representação conceitual, como em um nível mais baixo, no caso da coleta de características em nível de *hardware* ou sistemas operacionais. A coleta de dados em nível de sistema dá informação sobre as mudanças de estados e comportamentos no sistema de arquivos e as chamadas de programas realizadas a partir do sistema operacional. As abordagens CDE (*Code, Data and Environment*) [83], ReproZip [2], noWorkflow [84], PASS (*Provenance Aware Storage Systems*) [1] são alguns exemplos de soluções para coleta de dados em nível de sistema. Essas abordagens serão discutidas nas próximas seções.

3.1.1 noWorkflow (*Not Only Workflow*)

O *noWorkflow* (*Not-Only Workflow*) é uma abordagem proposta por [84] que faz a captura e gerência da proveniência de *scripts* usados na execução de aplicações científicas. O objetivo é identificar as informações detalhadas sobre o controle de fluxo das informações e as dependências de bibliotecas de *softwares* científicos invocados por meio de *scripts*. O *noWorkflow* dá suporte para o cientista explorar a proveniência capturada através dos mecanismos de consulta e visualização, permite fazer a deputação da execução, e ainda permite a reprodução.

O *noWorkflow* executa 3 etapas distintas para o tratamento dos dados de proveniência [84]: captura, armazenamento e análise. Na etapa de captura, todas as definições de funções do usuário, chamadas de funções, argumentos e variáveis globais referenciadas são representadas em árvores sintáticas abstratas, para que sejam associados com o *script* executado. As dependências de ambientes e dos módulos também são obtidos, assim como outros elementos durante a execução, tais como acesso a arquivos. A etapa de armazenamento registra a proveniência em disco em um subdiretório próprio dentro do diretório do *script* executado. Os dados estruturados são armazenados em banco de dados relacional e os dados não estruturados em arquivos. A ligação entre os dados estruturados e não estrutuados é identificada por meio de *hash* usando o algoritmo SHA1. A etapa de análise da proveniência aplica técnicas para verificar questões relacionadas a reprodução dos resultados. A abordagem faz três tipos de análise: baseadas em grafos, diferenças e consultas.

A abordagem *noWorkflow* tem o objetivo contornar uma deficiência da fata de coleta de proveniência de experimentos que adotam *scripts* para invocar a execução de *softwares* científicos. Mesmo dando um apoio importante a este cenário, o cientista ainda não tira as vantagens da orquestração e gerência automatizadas obtidas com a adoção dos SGWfCs. A abordagem coleta a proveniência e aplica análises sob esses dados auxiliando a validar e verificar diferentes execuções e, consequentemente, permite a avaliação de reproduções. Porém, não trata as questões relacionadas a portabilidade do ambiente para compartilhar a reprodução com outros cientistas, consequentemente, não fornece a infraestrutura computacional para a reprodução.

3.1.2 *Provenance Aware Storage Systems* (PASS)

O PASS [1] é uma abordagem para coleta e processamento de proveniência em um ambiente implantado em VMs com o *hypervisor* Xen [1]. Com o PASS, um experimento é implantado e executado em um conjunto de VMs convidadas executadas sob um sistema operacional hospedeiro. O mecanismo de proveniência do PASS fica implantado no sistema hospedeiro. O PASS intercepta as chamadas de sistemas

[1] http://www.xenproject.org/developers/teams/hypervisor.html

realizadas pelas VMs convidadas ao hospedeiro [1], monitorando as alterações nos sistemas de arquivos das VMs e armazenando essas informações em uma base de proveniência. A arquitetura e o contexto do PASS são mostrados na Figura 3.1.

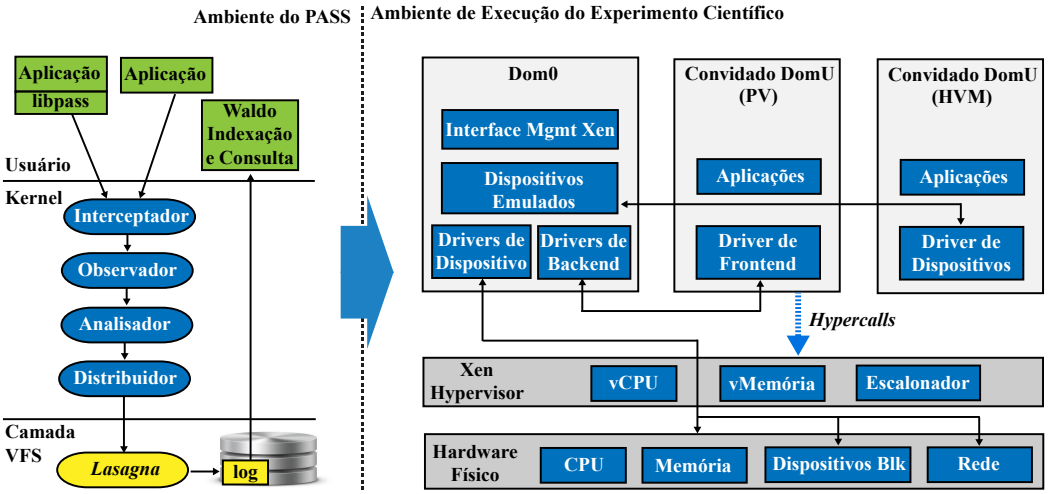


Figura 3.1: Arquitetura de captura de proveniência do PASS (adaptado de [1]).

O contexto de implantação do PASS segue as características do ambiente de virtualização Xen. Esse ambiente é composto por múltiplas VMs denominadas domínio (Dom). Uma das VMs, denominada Dom0, tem a função de centro administrativo. Ele é responsável pela inicialização (*boot*) e comunicação das VMs com o *hardware*. As demais VMs, denominadas visitantes, são nomeadas como DomU. As aplicações que compõem o experimento ficam implantadas nos sistemas convidados DomU.

Toda chamada de sistema feita a partir das VMs convidadas são interceptadas pelo módulo *interceptor*. Ele extrai as informações das estruturas de dados do *kernel* e passa para o módulo *observer*. O *observer* traduz as chamadas em registros de proveniência, criando relações de dependências entre arquivos e processos. Em seguida, o *analyzer* elimina os registros duplicados. O módulo *distributor* armazena a proveniência dos objetos em um *log* no sistema de arquivos próprio, denominado *Lasagna*, até que eles possam ser persistidos na base de proveniência. O último componente, denominado *Waldo*, obtém os registros de proveniência do *log* e armazena na base de proveniência, para posterior consulta e recuperação das informações.

O PASS é usado para a consulta de proveniência sobre detalhes de baixo nível e trilhas de execução seguidas durante a execução. É uma boa solução para aplicações que precisam de detalhes de baixo nível para a validação e reprodução. Porém, em muitos casos esse nível de detalhe não é necessário, causando um armazenamento excessivo de proveniência que não será posteriormente utilizada. Existem problemas significativos que devem ser observados no caso de abordagens que coletam a proveniência em nível de sistema [1]. Esse sistemas devem ser testados toda vez que o

kernel do sistema operacional sofre uma modificação, e portanto, é preciso verificar se as mudanças implantadas afetaram o mecanismo de coleta de proveniência.

3.1.3 *Code, Data and Environment* (CDE)

O desenvolvimento do CDE foi motivada pela barreira técnica de distribuição de código científico. O CDE é um mecanismo de implantação automática de código fonte, dados e ambientes usados para a execução de um conjunto de programas *x86-Linux* em outras máquinas *x86-Linux.* O CDE elimina a necessidade de instalação de *software* de um ambiente de origem para um de destino [85]. O CDE monitora as chamadas de sistemas a arquivos, códigos, dados e variáveis de ambiente durante a execução de um programa usando a ferramenta de depuração *ptrace* [2] [3].

O *ptrace* é um mecanismo de monitoramento que cria um processo denominado *tracer* que é responsável por observar e controlar a execução de um outro processo chamado *tracee*. O *tracer* observa e analisa a trilha de chamadas de sistemas na execução do *tracee*, permitindo capturar informações sobre o processo, usos de registradores e memória do processo *tracee*. O monitoramento ocorre após o número do PID (*Process Identifier*) do *tracee* ser anexado ao *tracer* através do comando *ptrace*. O CDE encapsula a ferramenta *ptrace* na sua implementação para permitir a criação de um pacote com todos os elementos invocados durante a execução de um aplicativo ou de um comando em um sistema operacional Linux.

A Figura 3.2 mostra o funcionamento do CDE. Neste contexto, é feita uma chamada para a execução de um aplicativo na linguagem Python [4] denominado *analise.py*. Nele é passado o arquivo *seq1.fasta* como parâmetro. A chamada de um aplicativo deve ser feita por meio do comando *cde* precedendo o comando de execução. Nesse passo, o CDE é invocado para capturar e encapsular os elementos usados na execução em um diretório, para armazenar os elementos e dependências identificadas pelo *ptrace*. Cada um desses elementos são copiados para o diretório do pacote CDE (*cde-package*) para serem compartilhados.

Para reproduzir uma pesquisa, basta obter o pacote CDE do autor do experimento e executar o comando $cde - exec$ para desempacotar o diretório do experimento com os elementos usados na execução. A partir desse ponto, o cientista pode navegar pelo experimento e reproduzi-lo segundo o roteiro original [83].

A abordagem possui três limitações [85]. A primeira está relacionada com incompatibilidade dos arquivos binários do pacote gerado pelo CDE em relação ao *kernel* ou o *hardware* da arquitetura Linux. Nesse caso somente a virtualização ou emulação podem superar essa limitação [83]. A segunda é quanto a mudança de

[2] http://linux.die.net/man/2/ptrace
[3] http://www.linuxjournal.com/article/6100
[4] https://www.python.org/

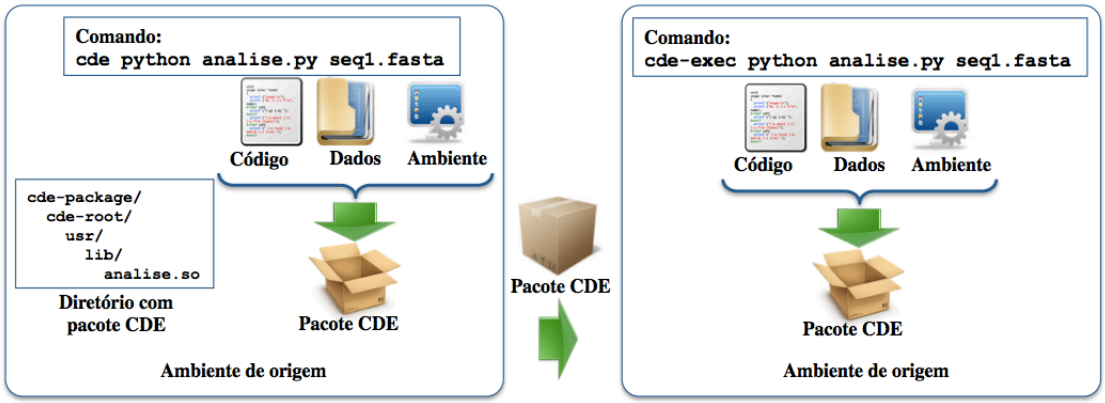


Figura 3.2: Esquema de funcionamento da abordagem CDE.

argumentos ou dados de entrada. Tais mudanças podem fazer um programa carregar novas bibliotecas que não estavam presentes na execução original. A última limitação é relativa a modificação dos argumentos ou dados de entrada que podem fazer o programa carregar outro arquivo que não está no arquivo.

O CDE apresenta-se como uma boa solução para o encapsulamento de um experimento permitindo a portabilidade entre ambientes computacionais. O CDE não é uma solução construída especificamente para um ambiente científico, portanto, não emprega conceitos e ferramentas de gerenciamento de experimentos. A abordagem não possui um mecanismo para a coleta de dados de proveniência, que são imprescindíveis para auxiliar na interpretação dos elementos usados na reprodução.

3.1.4 Reprozip/Vistrails/CrowLabs

O Reprozip é uma ferramenta para o empacotamento dos elementos necessários para reproduzir os resultados obtidos em uma pesquisa científica orientada pelos dados de proveniência [2]. O principal objetivo é o compartilhamento do experimento com os revisores e leitores para que os mesmos possam descompactá-lo e reproduzi-lo sem a necessidade de instalar *softwares* adicionais. O Reprozip combina o CDE [83] com o uso de VMs para gerar *snapshots* do computador. Ele encapsula somente os elementos que foram usados na execução do experimento. Trata-se de uma abordagem proposta em conjunto com o SGWfC Vistrails [31] aplicado ao ambiente científico para tornar os *workflows* portáveis e reprodutíveis na infraestrutura de terceiros.

O Reprozip executa dois estágios principais, o empacotamento do ambiente e o posterior desempacotamento em um computador de destino. A Figura 3.3 mostra os passos de empacotamento composto por 3 atividades: (1) captura de dados de proveniência, (2) análise da proveniência e (3) geração de pacotes. Na captura, o Reprozip usa o *SystemTrap* para obter a proveniência. Ele instrumenta e captura a trilha de execução das chamadas de sistema dos processo envolvidos na execução.

A proveniência é armazenada em uma base de dados gerenciada por um SGBD Mongo DB [5]. Em seguida, na análise, é criada uma árvore da proveniência do experimento. Cada nó dessa árvore representa um dado processo do sistema operacional e as arestas representam a ligação entre processo pai e os processos filho, permitindo identificar as dependências de recursos com programas executáveis, arquivos de entrada e saída. Ao final, é criado um pacote com o *workflow*, *softwares* e os arquivos de entrada e saída usados na derivação dos resultados.

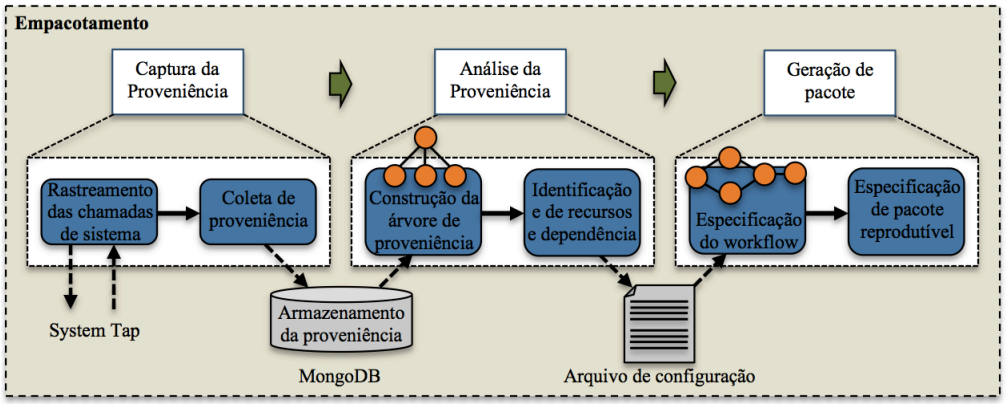


Figura 3.3: Processo de criação de um pacote Reprozip [2].

No processo de desempacotamento extrai o pacote do experimento em um determinado diretório e faz a configuração dos programas, conjuntos de dados, parâmetros de configuração e variáveis de ambiente do *workflow* [2]. Como o Reprozip utiliza a abordagem CDE para prover o empacotamento dos códigos, dados e ambiente, consequentemente, ele herda todas as limitações da abordagem. Apesar de trabalhar com *workflows* científicos e dados de proveniência, a abordagem não prevê as questões e peculiaridades de experimentos que usam ambientes especiais como *clusters* ou nuvem computacional. Além disso, o modo de depuração inclui uma carga de atraso significativa da execução do experimento.

3.2 Artigos Executáveis

Os esforços na área de artigos executáveis tem o objetivo de tornar experimentos científicos e seus resultados, na forma de artigo científico, reprodutíveis a leitores, revisores e demais interessados. Os sistemas Collage [3], SHARE [86], Paper Mâché [4], IODA [87], Madagascar [61] e Sweave [88] são alguns exemplos de abordagens nessa linha, os quais terão a sua descrição, arquitetura, principais características e o estado atual de desenvolvimento apresentados nas próximas seções. Com exceção do Sweave e do Madagascar, os demais sistemas foram selecionados como finalistas no Desafio de Artigos Executáveis da Elsevier [11].

[5] http://www.mongodb.org/

3.2.1 *The Collage Authoring Environment*

O Collage é uma infraestrutura de *software* que permite o desenvolvimento e publicação colaborativa dos trabalhos científicos na forma de artigos executáveis [3]. Ele permite a inserção de pedaços de códigos executáveis na estrutura de um artigo para permitir a reprodução dos resultados. Esses fragmentos permitem a definição das entradas de dados e execução da computação vinculada a esse fragmento, permitindo um acompanhamento de uma visualização interativa dos resultados relacionada a esse fragmento. A visão conceitual de um artigo executável e o esquema de funcionamento do ambiente Collage são apresentados na Figura 3.4.

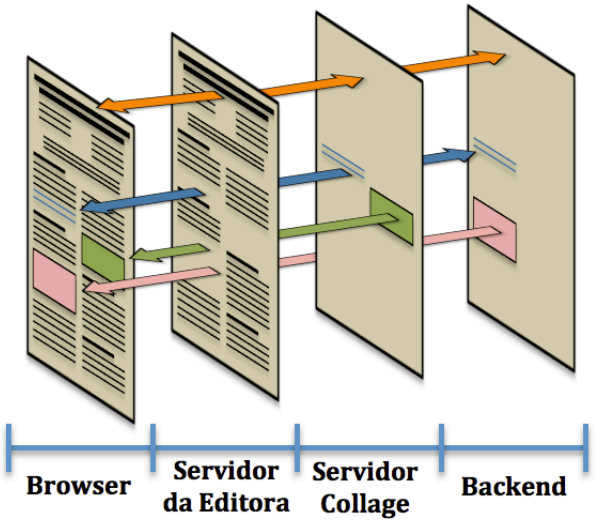


Figura 3.4: Estrutura de um artigo executável no Collage (adaptada de [3]).

O artigo executável no Collage possui quatro elementos [3]: conteúdo estático (seta laranja), dados primários (seta azul), entrada de dados (seta vermelha) e visualização interativa dos resultados (seta verde). O conteúdo estático é composto pelos elementos que não podem ser alterados, tais como textos e algumas imagens. Ele é obtido a partir do servidor de aplicação da editora responsável pela publicação. A função do conteúdo estático é estendida pelos elementos interativos, que permite o acesso aos dados primários e a reexecução de novas entradas de dados para verificar os resultados produzidos na visualização interativa. Os elementos interativos são obtidos a partir do servidor de aplicação do Collage. Os dois últimos elementos podem estar em locais externos e podem ser referenciado pelo servidor Collage.

Em termos de executabilidade, o Collage permite informar dados nos formulários do artigo para testar novos valores e avaliar os resultados produzidos, por meio da inserção de códigos executáveis embarcados na estrutura do artigo [3]. A abordagem permite a compatibilidade com diversas linguagens de programação e de *scripts* para a construção dos fragmentos executáveis. A validação dos resultados é feita de forma manual, através da reexecução dos trechos de códigos embarcados. Para a validação,

o usuário informa o novo conjuntos de dados de entrada ou parâmetros, reexecuta o trecho de código e avalia se os resultados produzidos são os esperados.

O sistema de licenças do Collage controla o acesso aos dados primários através de um esquema de registro *(login)*. Não são tratadas questões referentes a licenças e direito de cópia do sistema operacional e de *softwares* aplicativos. O Collage permite o acesso a recursos remotos por meio de *links* que referenciam componentes externos vinculados a serviços de computação e de armazenamento de dados. A abordagem não conta com o apoio de ferramentas de gerenciamento de *workflows*. Além disso, não são tratadas as questões relacionadas a proveniência dos dados gerados com a reprodução. A abordagem prevê o uso de arquiteturas especiais como a computação em nuvem, porém, esse recurso ainda não foi implementado na plataforma.

3.2.2 *Sharing Hosted Autonomous Research Environments*

O SHARE , desenvolvido por Van Gop & S. Mazanek [86], é um portal *web* que permite a criação e compartilhamento de artigos científicos executáveis. O seu desenvolvimento foi motivado por alguns fatores, tais como, a dificuldade na instalação ou configuração para a execução das computações mediante as entradas de dados, os problemas para a aquisição de determinadas versões do *software* para a execução do conjunto de funções, e o fato de pesquisas científicas combinarem um conjunto de *softwares* para a resolução de um problema, demandando *download*, instalação e configuração de cada *software* desse conjunto, assim como a necessidade de controle e distribuição das licenças desse *software* de uma forma adequada.

A proposta do SHARE é que os leitores tenham acesso ao ambiente usado no desenvolvimento de uma pesquisa e a todos os objetos de pesquisa (*software*, dados, infraestrutura e ambiente operacional), devidamente instalados e configurados. A abordagem propõe associar um artigo digital com uma VM, criada, configurada e armazenada dentro da infraestrutura onde o SHARE está implantado. O SHARE abrange 3 tipos de usuários [86]: autores, editoras e leitores. A Figura 3.5 apresenta as atividades executadas por cada usuário. Os autores criam VMs SHARE e associa os seus elementos a um artigo. Um artigo é apresentado aos editores e leitores por meio de um *web browser*, onde cada um dos elementos reprodutíveis está vinculado a um *link* que dá acesso a uma VM. Os editores possuem uma área especial vinculada aos pacotes de experimentos (*bundle*) SHARE para acessar as VMs diretamente.

São apontadas algumas características que orientaram o desenvolvimento de um artigo executável [86]. A executabilidade das VMs dão a capacidade de explorar as equações, tabelas e gráficos interativos. A compatibilidade a curto e longo prazo possui um gargalo relacionado com a duração do *hypervisor*, atualmente, sob a

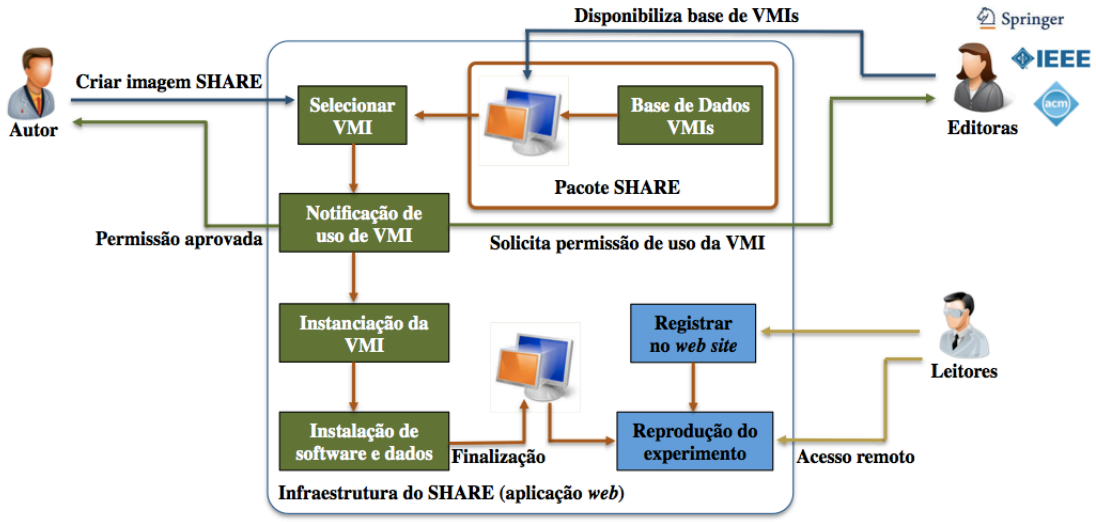


Figura 3.5: Processo de reprodutibilidade de VMs com o SHARE.

versão acadêmica do Oracle Virtual Box[6]. A validação é feita manualmente, com a reexecução da VM para verificar a reprodução dos resultados. O licenciamento e direitos de cópia sob os dados, *softwares* e sistema operacional são controladas por restrição de uso da VM para cada sessão do usuário durante um tempo pré-definido.

Em termos de sistemas, as VMs são gerenciadas no servidor SHARE evitando que os usuários tenham que gerenciar a infraestrutura do servidor. A infraestrutura de VMIs do SHARE permite a concepção de experimentos que usam estratégias de HPC, desde que os recursos do servidor sejam disponíveis e compartilhados para a construção dos experimentos do artigo. O SHARE armazena proveniência das sessões das VMs e das VMIs utilizadas como base para sua criação sob a forma de *logs*. A abordagem não utiliza modelos e padrões como o OPM e o W3C Prov para representação dos dados e informações de proveniência. O controle e otimização dos arquivos com grande volume de dados é feito dentro da plataforma do SHARE, evitando a obtenção e armazenamento dos dados nas estações de trabalho local.

3.2.3 Paper Mâché

O Paper Mâché é um sistema de gerenciamento que permite explorar artigos científicos de um forma interativa, reproduzindo e validando os resultados para testar as hipóteses [4]. Ele permite criar um artigo e submetê-lo aos revisores, para que eles possam explorar e avaliar antes da publicação. Após publicado, o sistema permite explorar, discutir e enviar os comentários sobre os artigos para os autores. A Figura 3.6 apresenta a arquitetura geral com os seus três elementos: (1) o artigo executável, com conteúdo textual, figuras, áudio e vídeos, (2) os comentários, contendo a re-

[6]http://www.oracle.com/technetwork/pt/server-storage/virtualbox/downloads/index.html

visão e discussão sobre o material publicado e compartilhado, e (3) a VM, com os códigos-fonte, arquivos executáveis, dados, bibliotecas e dependências encapsulados.

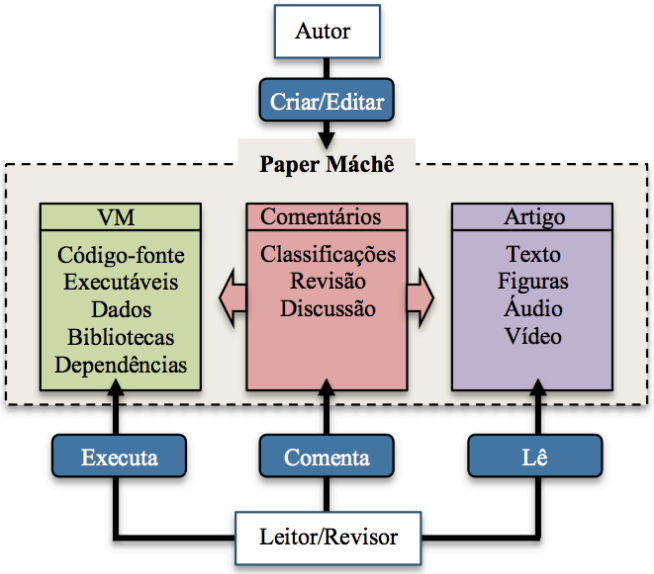


Figura 3.6: Estrutura geral do Paper Máchê [4]

.

Esse conjunto de elementos são utilizados pelos autores para a criação e modificação de um artigo científico executável e da sua respectiva VM. O conteúdo dos artigos são criados em editores de texto tradicionais, nos formatos *.doc ou *.tex. A criação da VM relacionada ao artigo é feita por uma ferramenta de criação de VMI por meio de um *hypervisor* definido pelo Paper Mâché. O autor deve criar os *scritps* na plataforma do sistema para relacionar o conteúdo publicado no artigo com os componentes que estão encapsulados na VM. Essa associação permite recuperar e reexecutar os elementos executáveis do artigo. Após a submissão, é possível ler o artigo eletrônico, executar as computações necessárias para validá-lo, e ainda pode criar um conjunto de comentários sobre a experiência com o artigo executável tendo a capacidade de interagir com o autor através dessa funcionalidade.

A executabilidade foi desenvolvida a partir dos *scripts* que ligam os elementos do documento digital com os componentes na VM. A compatibilidade a curto e longo prazo são preservadas através das VMs. O ambiente do experimento fica comprimido dentro da VM e o *hypervisor* usado pela abordagem deve garantir que essa VM possa ser recuperada, reimplantada e reexecutada. O processo de validação é feito manualmente, ou seja, é preciso reexecutar e testar novos valores sob os componentes executáveis do artigo e verificar se os resultados esperados foram produzidos. O licenciamento e direitos de cópia sob os dados, *softwares* aplicativos e sistema operacional são controlados pelo autor. Ele implanta e compartilha na VM somente os elementos que ele permite que terceiros tenham acesso. Os *softwares* proprietários

de terceiros são publicados na VM sob a sua responsabilidade.

O armazenamento da proveniência e o gerenciamento das trilhas de execução não foram descritos na abordagem. A característica de gerenciamento e compartilhamento de grandes volumes de dados limita-se a capacidade de tamanho suportado pela VM. Foram ressaltadas a necessidade de expandir a solução para a computação em nuvem, implantando os experimentos em VMs na infraestrutura dos provedores, porém, essa solução ainda não foi adotada pela abordagem. A adoção da nuvem, segundo os autores, podem dar a capacidade de testar e interagir com experimentos na escala de supercomputadores, o que não é possível com a atual solução.

3.2.4 *Interactive Open Document Architecture* (IODA)

O IODA é uma arquitetura que permite a representação de um artigo científico eletrônico como uma estrutura de objetos digitais multicamadas [87]. O artigo pode ser construído em diversas camadas de representação usando os serviços e ferramentas atualmente existentes na *web*. O IODA fornece uma estrutura de arquivso XML que permite ligar os conteúdos de cada uma das camadas, relacionando, por exemplo, imagens e gráficos com os fragmentos de códigos responsáveis por criá-las.

Os documentos criados no IODA possuem três camadas [87]: dados, informação e conhecimento. A camada de dados registra os fatos apresentados no artigo. Essa camada possui um documento principal que faz referência a um conjunto de arquivos, fragmentos de códigos, *scripts* e etc. A camada de informação possui os padrões de interpretação que são construídos sob os componentes da camada de dados. Essa camada combina os objetos do conteúdo de um documento (tipos) com as estruturas sintáticas (visão) do *layout* do documento, e com os fragmentos da imagem de um documento (marcações). A camada de conhecimento apresenta o contexto dos padrões de interpretação baseados na camada de informação. O objetivo dessa camada é auxiliar o usuário a interpretar as informações apresentadas no documento. Essa camada é composta em geral por um conjunto de anotações e *links*.

A abordagem IODA é composto por 3 etapas [87]: submissão, revisão e publicação. A submissão envolve a construção do documento principal com a composição das camadas de dados, informação e conhecimento com os objetos do texto e anexos já consolidados e prontos para a revisão. A revisão é um processo interativo que cria várias versões do documento principal, uma versão para cada revisão. Ela pode resultar na modificação das camadas de dados e informação. A última fase, de publicação, disponibiliza o artigo executável para o público em geral contendo os elementos usados para a produção das tabelas, equações, gráficos e etc.

O IODA provém a característica de executabilidade através da seleção do componente seguida da sua interpretação, por exemplo, é possível selecionar uma equação

e ter acesso ao conjunto de dados usado por essa equação para a produção dos resultados. Também é possível associar os conjuntos de dados com serviços previamente registrados para permitir que elementos mais complexos, tais como tabelas e gráficos, possam ser orquestrados para a geração dos resultados e formação desse objetos no artigo. A compatibilidade a curto e longo prazo do IODA é baseada na construção de componentes executáveis direcionados para *web browsers*, tornando os artigos executáveis compatíveis com diversos sistemas operacionais e arquiteturas de sistemas, assim como para futuros sistemas baseados em *web browsers*.

O processo de validação é feito de forma manual, através da reexecução dos componentes executáveis do artigo para verificar se os resultados publicados foram repetidos. Porém, é preciso solicitar o acesso a camada de dados ao autor para que seja possível fazer a reexecução dos componentes executáveis. A camada de informação auxilia os revisores reexecutar os fragmentos de código e, consequentemente, validar os resultados produzidos. O licenciamento e direitos de cópia não são tratados. O registro e rastreamento da proveniência são baseados nas informações contidas na camada de conhecimento da abordagem. A proveniência é baseada no armazenamento dos *log*, *links* e nas anotações feitas pelos autores. A abordagem não provém a infraestrutura de computação necessária para a composição dos experimentos e, consequentemente, não trata os casos que utilizam arquiteturas especiais.

3.2.5 Sweave

O Sweave é uma ferramenta que permite inserir códigos R para a análise de dados em documentos Latex [88]. Ele tem o objetivo de tornar os documentos dinâmicos, pois, ao invés do autor inserir objetos de imagens, gráficos ou tabelas no texto, ele inclui o código Sweave necessário para gerar tais objetos a partir de uma execução no R. O resultado de cada execução gera como saída novas imagens, gráficos e tabelas e os atualiza na região correspondente no documento Latex. Essas características de atualização de dados e inserção de códigos nos relatórios permitem que uma pesquisa torne-se reprodutível, pois a medida que os dados são inseridos ou atualizados, os resultados dessas modificações são automaticamente apresentados no documento.

O R é uma linguagem e um ambiente aplicado para a computação e gráficos estatísticos, sendo uma derivação construída a partir da linguagem e ambiente S [7]. O R fornece um conjunto de técnicas para cálculos estatísticos e geração de grafos que abrangem ferramentas para modelagem linear e não linear, testes estatísticos clássicos, análises de séries temporais, classificação, agrupamento etc. Diante desse conjunto de oportunidades oferecidas pelo R, o Sweave é definido como uma funcionalidade do R que é implementada por algumas funções do pacote *utils*.

[7] http://www.r-project.org/

O Sweave combina as duas etapas do projeto de análise de dados: a análise de dados propriamente dita, que usa *softwares* estatísticos para esta tarefa, e a apresentação dos resultados da análise, por meio de imagens, gráficos e tabelas [88]. As funções contidas no Sweave permitem a execução dessas duas tarefas, resultado na edição ágil de documentos combinando o Latex e o R. Com o Sweave, os códigos R juntamente com o conjunto de dados de entrada utilizados nas análises são integrados ao documento Latex tornando possível repetir os resultados, e modificá-los de forma a analisar novos comportamentos e resultados finais.

3.2.6 Madagascar

O Madagascar é um pacote de *software* aplicado a análises de dados multidimensionais e experimentos computacionais reprodutíveis [62]. Ele fornece um ambiente computacional para pesquisadores que trabalham com processamento de dados e imagens digitais na geofísica e áreas de pesquisa relacionadas. O Madagascar é uma ferramenta para transferência de tecnologia, onde registros de experimentos, denominados *computational recipes* podem ser verificados, trocados e modificados pelos usuários do sistema. Ele possui códigos que podem ser incluídos em documentos Latex para fazer a ligação entre as figuras inseridas no documento com os *scripts* responsáveis por gerá-las, permitindo a sua reprodução [61].

O Madagascar atende 3 atividades comuns de um experimento apoiado pela computação [61]: (1) implementa algoritmos numéricos, (2) apoia a condução de experimentos científicos, e (3) publica os resultados desses experimentos. O projeto Madagascar mantém um repositório público contendo os dados de pesquisa. Ele está dividido em 2 camadas principais: (1) camada de documentação que usa os arquivos HTML e PDF, e (2) a camada de fluxo de processamento. A Figura 3.7 apresenta a estrutura multicamadas do Madagascar.

Na camada de fluxo de processamento foi adotada uma ferramenta baseada na linguagem Python, denominada SCons (*Software Construction*) para gerenciar figuras geradas por *workflows* [61]. O SCons é um programa de linha de comando desenvolvido para a construção de *software*. Ele permite a configuração do ambiente de reprodução através de *scripts* python, e oferece um mecanismo de análise e resolução de dependências, permitindo que um ambiente de experimento de computação seja portável e reprodutível [5]. Os experimentos podem ser preparados com outras ferramentas, tais como, Matlab [8], Mathematica [9], Python dentre outros, criando-se uma nova camada que é invocada pelos *scripts* SCons. Na camada de documentação são estabelecidas ligações entre as figuras de um artigo e os códigos

[8] http://www.mathworks.com/products/matlab/
[9] http://www.wolfram.com/mathematica/?source=nav

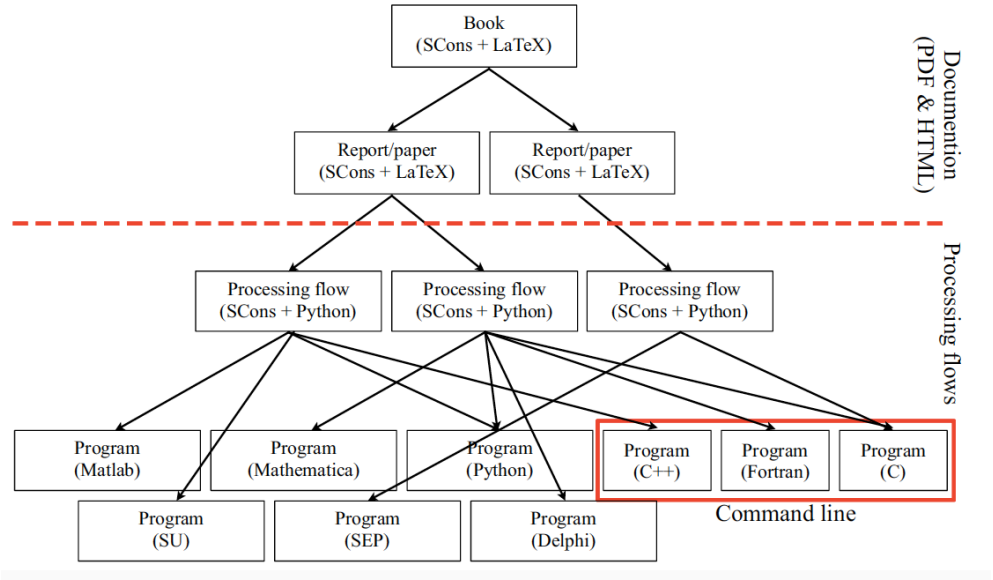


Figura 3.7: Estrutura multicamadas do Madagascar [5]

usados para gerá-las, feitas através da combinação do SCons com o Latex para a geração de arquivos PDF, HTML e MediaWiki [5].

3.3 Pesquisa Reprodutível na Nuvem

Diversos autores [89], [24], [23], [90] discutem as vantagens de migrar o ambiente dos experimentos das estações de trabalho, *clusters* e grades para a nuvem. As características e recursos da nuvem podem aumentar a capacidade de reprodução, motivando o desenvolvimento de abordagens baseadas neste paradigma de fornecimento de recursos. As abordagens WSSE [38], Chef [23] e AMOS [6] são as principais representantes dessa classe de mecanismos de reprodução e serão discutidos nas próximas seções.

3.3.1 *Whole System Snapshot Exchange* (WSSE)

A abordagem Troca Instantânea de Sistema é uma formulação de conceito desenvolvida para tratar as dificuldades de reprodução relacionadas a infraestrutura de computação e armazenamento de objetos de pesquisa. O WSSE propõe gerar *snapshots* digitais de dados e código fonte para serem produzidos e distribuídos dentro de um provedor de computação em nuvem [38]. A proposta é o WSSE utilizar as vantagens da implantação de experimentos científicos na nuvem com o suporte a reprodução por meio da definição de três camadas: dados, sistemas e serviços.

Na camada de dados, a nuvem pode armazenar e compartilhar grandes conjuntos de dados com recursos necessários para processar computações sobre esses dados.

Geralmente, os conjuntos de dados científicos podem variar de um tamanho de dezenas de gigabytes para vários terabytes ou mais, assim, mesmo tendo acesso a conexões de *internet* muito rápidas, em muitos casos, não é viável copiar esses grandes arquivos de dados de um computador para outro de uma forma fácil e eficiente [38]. Para evitar esse gargalo, o WSSE propõe que os dados sejam trocados exclusivamente dentro do ambiente de nuvem computacional.

Na camada de sistemas, os computadores usados na produção dos resultados são copiados em sua totalidade, incluindo o sistema operacional, *softwares* e banco de dados, para uma VMI que pode ser trocada com outros pesquisadores [38]. Esse processo permite que os pesquisadores sejam capazes de obter réplicas exatas do computador usado para a produção dos resultados publicados. A camada de serviço permite que as computações implantadas na nuvem possam ser acessadas por aplicações externas gerando uma solução de reprodução orientada a serviços. Os dados e sistemas vinculados aos experimentos científicos podem ser armazenados em VMIs na nuvem preservando o ambiente usado na execução do experimento original.

Foram identificados alguns desafios durante o seu projeto, principalmente em relação a questões de licença e distribuição de *softwares*, pois em geral, a licença é restrita a um computador. Além disso, devem ser consideradas questões relacionadas a coleta de dados de proveniência em nível de sistemas, pois a nuvem abstrai detalhes de infraestrutura, limitando as informações das camadas acima do nível de VM e sistema operacional. O WSSE ainda é um conceito, portanto, não existe uma implementação que aponte as características operacionais para o seu funcionamento.

3.3.2 Chef

Klinginsmith *et al.* [23] propõem uma abordagem denominada Chef para a construção de ambientes de *clusters* virtuais implantadas sob infraestruturas de nuvem. Na abordagem, os recursos computacionais são divididos em duas camadas: (1) infraestrutura e (2) *software*. A primeira gerencia a infraestrutura em nuvens IaaS, comunicando com a API do provedor de nuvem para instanciar VMs, configurar a rede e alocar espaço de armazenamento. A segunda, gerencia os *softwares* responsáveis pela execução dos comandos na VM. Ambas as camadas são gerenciadas pela ferramenta de gerenciamento de configuração Chef. O Chef é usado para automatizar a configuração dos *clusters* virtuais e para a instalação dos *softwares* necessários para a reexecução dos experimentos, possibilitando a reprodução do ambiente.

A abordagem Chef foi implantada em AMIs (*Amazon Machine Image*) na Amazon EC2 e EMIs (*Eucalyptus Machine Image*) FutureGrid's da nuvem do Eucalyptus[10]. Na abordagem é necessário instanciar, configurar e registrar as VMIs manual-

[10]https://www.eucalyptus.com/eucalyptus-cloud/iaas

mente, e ainda, instalar um cliente do Chef para auxiliar no processo de configuração dos *softwares* nas VMIs. O resultado final é a possibilidade de replicação dos *clusters* virtuais em diferentes provedores de nuvem utilizando uma camada de *software* em comum [23]. A arquitetura do Chef é formada por seis componentes [23]:

- Livro de Receitas (*Cookbook*): agrupa artefatos Chef (ex. receitas) relacionado uns com os outros;
- Receitas (Recipes): é uma unidade básica de configuração do Chef. Uma receita pode interagir com outras receitas e demais artefatos.
- Recursos (*Resources*): são usados pela receita para a execução de alguma ação;
- Servidor Chef (*Chef Server*): local que armazena todos os artefatos do Chef;
- Cliente Chef (*Chef Client*): máquinas clientes que o servidor Chef administra as receitas para instalar e configurar o software.
- Faca (*knife*): ferramenta de linha de comando de uso administrativo do Chef.

3.3.3 Reprodutibilidade em Grades com o AMOS

Strijkers *et al.* [6] apresenta um conjunto de ferramentas de *e-Science* no estado da arte que podem ser amplicadas para descrever códigos, *softwares* e os parâmetros de experimentos científicos para a geração de uma arquitetura para a concepção de artigos eletrônicos. É proposta a adoção de um conjunto de práticas para preservar as dependências em um artigo executável para permitir a reexecução de um experimento. São destacados o uso de *workflows* científicos como uma forma de descrever o método e executar os experimentos, e a computação em nuvem IaaS como uma plataforma para encapsular as dependências de código e *software* em VMs.

O Sistema AMOS, proposto por Strijkers *et al.*, usa um conceito de um *template* mínimo de VM contendo um conjunto de ferramentas de grades computacionais previamente instaladas para implementar um mecanismo que inicializa e configura VMs sob demanda [6]. O objetivo é que templates de VMs possam ser recriadas ou clonadas para a execução e reexecução de um *wokflow*. Com isso os pesquisadores podem criar diversos *templates* e armazená-los em uma base de dados de artigos executáveis. Os processos de gerenciamento dos dados e da execução da aplicação das VMs são instrumentados por um agente de *workflow* (WFA), ou seja, um SGWfC.

A Figura 3.8 apresenta uma proposta de arquitetura para a representação e reprodução de experimentos. O AMOS é incluído na arquitetura com três elementos: proveniência, gerenciador de infraestrutura virtual e URLs. A proveniência obtém as

informações da execução do *workflow*. O gerenciador de infraestrutura usa os recursos dos provedores de nuvens para a criar o ambiente de execução dos componentes do *workflow* baseado nos *templates* armazenados na base de artigos executáveis. Os conjuntos de dados usados na execução podem ser referenciados através de URLs

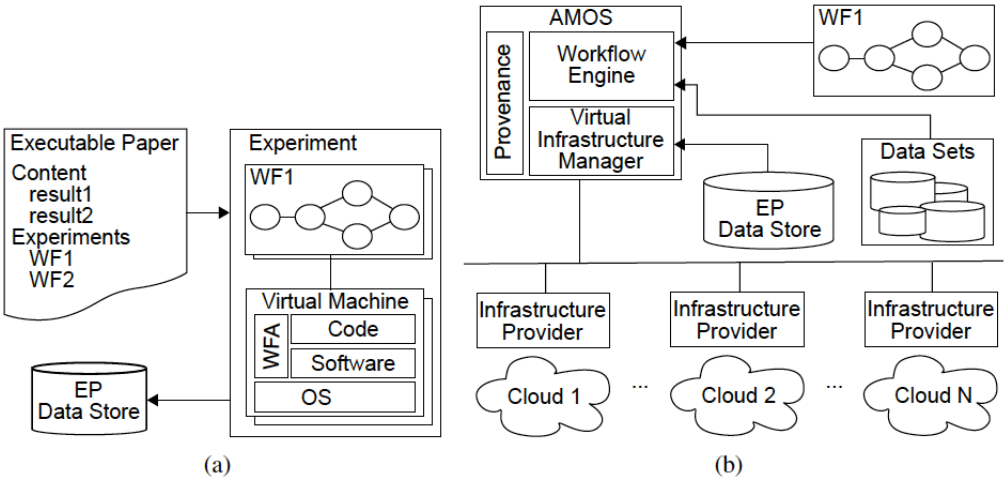


Figura 3.8: Proposta de arquitetura de descrição e reprodução de experimentos [6].

O AMOS apresenta a possibilidade de preservação de código e dados em VMs para a preservação da compatibilidade do ambiente experimental, porém, destacam que essa compatibilidade é dependente do *software* de virtualização. Os autores enfatizam que os problemas relativos a artigos executáveis não estão relacionados apenas com as características técnicas, mas envolve ainda a manutenção dos objetos de pesquisa usados no experimento. Os autores destacam ainda que não existe solução eficaz baseada em metadados e dados de proveniência que aumente a capacidade dos artigos executáveis, portanto, a chave seria a infraestrutura de virtualização.

3.4 Comparação das Abordagens

Os 10 fatores extraídos do mapa conceitual, apresentado na Seção 2.5 orientaram na análise e comparação das abordagens apresentadas neste capítulo. As características relacionadas com o 10 fatores podem ser representados da seguinte forma: (1) metodologia, englobando a RepeTição, ReproDução, ConFirmação, AuDitoria e ReVisão $\{RT, RD, CF, AD, RV\}$, (2) tipos de dados, identificando os dados PRimários, INtermediários, FInais, MetaDados, ProVeniência $\{PR, IN, FI, MD, PV\}$, (3) agentes, contendo as pessoas que têm acesso a uma pesquisa, AuTor, ReVisor, EDitor, LeiTor $\{AT, RV, ED, LT\}$, (4) apresentação, sendo de duas formas Artigo Executável e Artigo Tradicional $\{AE, AT\}$, (5) avaliação, com atividades de VAlidação e VErificação $\{VA, VE\}$, (6) licenciamento, podendo ser PUblico, PRivado e MIsto $\{PU, PR, MI\}$, (7) *hardware*, considerando os recursos de ARmazenamento e CoM-

putação $\{AM, CM\}$, (8) Documentação, representada pelas AnoTações e MaNuais $\{AT, MN\}$, (7) configurações, com as informações sobre as Variáveis de Ambiente e Parâmetros de Configuração $\{VA - PC\}$, e (10) *software*, que pode ser representado pelos *softwares* APlicativo ou BAsico, BIbliotecas, WorkFlow, ScRipt e Código Fonte $\{AP, BA, BI, WF, SC, CF\}$.

As Tabelas 3.1 e 3.2 apresentam a avaliação das abordagens do estado da arte em relação à proposta de reprodução apresentada neste trabalho. A análise foi orientada segundo os fatores extraídos do mapa conceitual proposto na Seção 2.5. Para cada um dos 10 fatores é assinalada uma ou mais siglas que definem o recurso tratado pela abordagem. Essas siglas foram definidas na Seção 2.5. Existem os casos na tabela em que as informações sobre um dado fator de uma abordagem não são mencionadas na bibliografia, portanto, foi assinalada com o símbolo '$-$' significando "desconhecido".

Tabela 3.1: Abordagem de avaliação baseada nos fatores apresentados na taxonomia.

Abordagens Baseadas em Proveniência					
Abordagem	**Atores**	**Tipos de Dados**	**Documentos**	**Avaliação**	***Hardware***
Cde	AT-LT	PR-IN-FI	-	VE	CM
Pass	AT-LT	PR-IN-FI-PV	-	VE-VA	CM-AM
noWorkflow	AT-LT	PR-IN-FI-MD-PV	-	VA-VE	-
Reprozip	AT-RV-ED-LT	PR-IN-FI-MD-PV	AT	VE-VA	CM-AM
Abordagem de Artigos Executáveis					
Collage	AT-LT	PR-FI	-	VE	-
Ioda	AT-RV-LT	PR-FI-PV	AT	VE	-
Madagascar	AT-LT	PR-FI	AT-MN	VE	-
Paper Máchê	AT-RV-LT	PR-IN-FI	-	VE	CM-AM
Share	AT-ED-LT	PR-IN-FI-PV	-	VE-VA	CM-AM
Sweave	AT-LT	PR-FI	-	VE	-
Abordagens com Computação em Nuvem					
Wsse	AT-LT	PR-IN-FI	-	VE	CM-AM
Chef	AT-LT	PR-IN-FI	-	VE	CM-AM
AMOS	AT-LT	PR-PV	-	-	CM-AM

Tabela 3.2: Abordagem de avaliação baseada nos fatores apresentados na taxonomia.

Abordagens Baseadas em Proveniência					
Abordagem	**Licenciamento**	**Metodologia**	**Apresentação**	**Parâmetros**	***Software***
Cde	PU	RD	-	VA-PC	AP-BA-BI
Pass	PU	RD	-	VA-PC	AP-BA-BI
noWorkflow	PU	RD	-	VA-PC	AP-BA-BI
Reprozip	PU	RD	AT	VA-PC	AP-BA-BI-WF-SC-CF
Abordagens de Artigos Executáveis					
Collage	PU-PR	RD	AE	PC	SC-CF
Ioda	-	RD	AE	-	SC-CF
Madagascar	PU	RD	AT	PC	BI-SC-CD
Paper Máchê	-	RD	AT	VA-PC	AP-BA-BI-SC-CF
Share	PU	RD	AT	VA-PC	AP-BA-BI-SC-CF
Sweave	PU	RD	AT	PC	BI-SC-CF
Abordagens com Computação em Nuvem					
Wsse	-	RD	-	VA-PC	AP-BA-BI-SC-CF
Chef	-	RD	-	VA-PC	AP-BA-BI-SC-CF
AMOS	-	RD	AT	VA-PC	AP-BI-SC-CF

3.5 Considerações Finais

Foram apresentadas as diferentes abordagens, esforços e tecnologias para permitir a publicação e compartilhamento de objetos científicos. Em termos de publicação dos experimentos, foi possível identificar os benefícios dos provedores de nuvens pública para possibilitar o compartilhamento e implantação da infraestrutura de uma pesquisa científica reprodutível. Foram listadas ainda algumas vantagens de adotar esse paradigma, principalmente, para soluções que usam ambientes HPC.

Diversas abordagens para a coleta e armazenamento de metadados e dados de proveniência para a reprodutibilidade. Elas obtém os dados em nível de sistema operacional, tais como as informações sobre o *software* e bibliotecas, além do conjunto de arquivos usados e produzidos durante a execução do *workflow* científico dos experimentos. Porém, são previstos experimentos executados em apenas um sistema de computador, ou seja, nenhuma das soluções para reprodução em nível de sistemas aborda o nível de *hardware*, pelo menos em termos de perfil de componentes utilizados (processador, memória e disco), ou mesmo a quantidade de nós de processamento envolvidos.

Grande parte das soluções não possuem mecanismos para a avaliação dos resultados, a analise é feita de forma não automatizada observando o resultado da reprodução e comparando com a da execução. Avaliando esse conjunto de trabalhos, foi identificada a necessidade de conceber formas de avaliação da reprodutibilidade e da qualidade dos resultados alcançados para confirmar o processo de reprodução. Algumas soluções desenvolvem o registro de metadados e proveniência para manter um histórico de derivação dos resultados, porém, essa adoção não foi unânime.

Um dos fatores que limitam a reprodução pesquisas em ambientes especiais é o acesso a infraestrutura necessária para a reprodução. O acesso a infraestrutura pode ser um fator determinante inviabilizando a reprodução do experimento. Nenhuma das abordagens trata esse fator, algumas ainda criam uma certa dependência, pois definem que a reprodução é dependente do ambiente no qual uma pesquisa foi concebida. Portanto, os recursos ficam limitados a capacidade oferecida pelo proprietário da abordagem de reprodução.

As abordagens apresentadas não tratam as questões relacionadas ao acesso e armazenamento de dados em ambientes paralelos e distribuídos. Experimentos da *e-Science* prevêem soluções que acessam a grandes conjuntos de dados e a ambientes distribuídos. Desta forma, uma abordagem de reprodução deve estar preparada para lidar com as peculiaridades desses tipos especiais de processamento. As poucas abordagens que trataram ambientes de computação paralela edistribuída limita a reprodução a sua própria infraestrutura, não permitindo a portabilidade da pesquisa para uma infraestrutura de computação diferente.

Capítulo 4

Metodologia

Este capítulo apresenta as ferramentas e os métodos empregados em cada uma das etapas da metodologia no desenvolvimento da abordagem ReproeScience. Essas etapas serão discutidas em 7 sessões. Inicialmente, na Seção 4.2 será feita uma formulação do problema de reprodutibilidade aplicado ao contexto de experimentos computacionais. A Seção 4.1 apresenta uma visão geral da abordagem, com uma síntese das 4 principais tecnologias empregadas no fornecimento da infraestrutura para a reprodução. A Seção 4.3 aborda a arquitetura do ReproeScience, com seus módulos internos e a interação com os componentes externos e demais tecnologias. A Seção 4.4 apresenta o modelo de dados usado para armazenar a proveniência. A Seção 4.5 mostra as notações e funções para a representação da proveniência com o W3C Prov e construção do documento XML adotando o padrão. A Seção 4.6 mostra o ciclo de vida com os algoritmos e detalhes da implementação dos módulos da abordagem. Ao final, na Seção 4.7 serão apresentadas as considerações finais.

4.1 Definição da Abordagem

O ReproeScience é uma abordagem de *software* que automatiza o monitoramento e preparação de uma pesquisa computacional reprodutível de forma que seja possível publicar, compartilhar e reproduzir em provedores de nuvem computacional. O ReproeScience foi desenvolvido para tratar experimentos projetados na forma de *workflows* científicos que usam ambientes de computação de alto desempenho, portanto, ele trata questões relacionadas a acesso a infraestrutura e preparação dos recursos de computação e armazenamento na nuvem para dar suporte a reprodução de experimentos dessa natureza.

Este cenário de apoio a reprodução de pesquisas computacionais é composto por três atores bem definidos: (1) o autor do experimento, que é responsável pelo desenvolvimento, publicação e compartilhamento da pesquisa reprodutível, (2) o revisor, que a partir de um acesso especial a pesquisa reprodutível, faz a devida

avaliação dos métodos, artefatos e resultados, e (3) os leitores, que são aqueles que terão amplo acesso a análise do artefatos publicados e reprodução da pesquisa com os recursos dos provedores de nuvem.

O ReproeScience obtém todas as características e comportamento da execução de um *workflow* científico para auxiliar o cientista a implantar a infraestrutura de um experimento computacional em um provedor de nuvem para permitir a publicação, compartilhamento e reprodução de uma pesquisa reprodutível. O componente possui um conjunto de métodos para gerenciar os agentes, atividades e entidades que compõem o *workflow*, assim como todos os relacionamentos entre esses elementos, que são a base do modelo de dados PROV-DM. A abordagem usa 4 tecnologias base para prover a infraestrutura de reprodutibilidade:

- O *Workflow* Científico e os Sistemas de Gerência de *Workflows* Científicos;
- A Proveniência de dados com os modelos OPM e W3C Prov;
- A Virtualização de Recursos; e
- A Computação em Nuvem.

Os *workflows* científicos e os SWfMS são usados para a representação e gerenciamento da execução dos experimentos científicos apoiados pelo ReproeScience. Os *workflows* são usados para documentar os atores, objetos de pesquisa e os processos que compõem a metodologia de execução do experimento. Eles descrevem conceitualmente e fisicamente os materiais, métodos e ferramentas que serão empregados dando uma visão geral do que será necessário para a execução.

Como os *workflows* são escritos em um SWfMS específico, é necessário incorporar o *software* SWfMS especializado para fazer o gerenciamento do *workflow*. A adoção das soluções de *workflows* foi motivada pelo fato de serem ferramentas especializadas na execução de *softwares* científicos, além de incorporar uma forma de documentação da metodologia, pois os *workflows* podem ser representados na forma de grafos permitindo uma representação gráfica do fluxo de execução.

Em geral, as aplicações científicas são orientadas a dados. Os cientistas executam experimentos computacionais para analisar o resultado final gerado pelo processamento do conjunto de programas que formam a metodologia. Em geral, o resultado final é formado por arquivos e dados derivados da execução de um conjunto de dados de entrada segundo determinados parâmetros de configuração informados em cada etapas de processamento. Os estados e comportamentos da execução em cada etapa formam o conjunto de variáveis responsáveis pela obtenção do resultado final. Essas variáveis são elementos importantes para a avaliação e interpretação dos materiais, métodos e ferramentas empregados.

Esse conjunto de informações podem ser armazenadas e gerenciadas sob a forma de metadados e dados de proveniência. Os metadados e dados de proveniência são elementos fundamentais para a verificação e validação dos experimentos. Eles orientam a implantação e recuperação da infraestrutura computacional para a reprodução. Somente os elementos informados na base de metadados e proveniência são reimplantados na infraestrutura para a reprodução.

A tecnologia de virtualização de recursos é usada para armazenar a infraestrutura dos computadores e demais elementos operacionais usados em um experimento computacional. Os principais objetivos da virtualização são preservação e portabilidade do ambiente operacional usado na produção dos resultados. Isto permite o compartilhamento, disponibilização (publicação) e transferência de uma pesquisa para um ambiente computacional de destino. A virtualização dá maior flexibilidade para a reimplantação da pesquisa, pois permite que outros usuários possam recuperar e reproduzir uma pesquisa apenas instanciando e executando a VM em que um experimento foi encapsulado. A virtualização agrega maior robustez no compartilhamento e transferência de um experimento.

O ReproeScience trabalha com o contexto de computação paralela e distribuída, por tal motivo, é previsto que um experimento pode executar um *workflow* científico em um conjunto de diferentes sistemas de computadores em cada execução. Para cada sistema de computador é criada uma VM diferente para representá-lo. Cada uma dessas VMs contém as suas próprias características operacionais, tais como, a quantidade e velocidade dos processadores, a capacidade de memória, largura de banda para o fluxo de informações na rede, arquitetura operacional, a capacidade do armazenamento em massa e sistema operacional. Esse conjunto de VMs possibilita a reimplantação de cada um dos sistemas de computadores usados na execução do *workflow* e, consequentemente, permite a reimplantação de toda a infraestrutura de reprodução do experimento.

O ReproeScience tem a computação em nuvem como o principal diferencial sob as outras abordagens de reprodutibilidade. Através da computação em nuvem é possível que o cientista publique a infraestrutura de sistemas de computadores utilizada no seu experimento, na forma de uma VM, e ainda permite que outros cientistas possam obter os recursos de computação necessários para fazer a reprodução no próprio local onde a sua infraestrutura foi publicada e está armazenada. Neste caso, é necessário apenas solicitar a implantação das VMs do experimento na infraestrutura do provedor de nuvem onde ele está hospedado. O ReproeScience possui ainda o diferencial de reimplantação e reprodução de pesquisas que utilizam ambientes especiais de computação paralela e distribuída. Esse suporte é possível devido à adoção de sistemas de gerenciamento de arquivos distribuídos unidos a grande e diversificada oferta de recursos computacionais dos provedores de nuvem.

Conforme abordado anteriormente, a computação em nuvem possui diferentes níveis de ofertas de recursos, distribuídos sob a forma de serviços. Estes recursos podem ser obtidos por um usuário mediante um contrato de fornecimento acordado com o provedor (SLA). No contexto do ReproeScience são obtidos serviços de infraestrutura de *data center* nos fornecedores de nuvem que usam o modelo de serviço IaaS. Este nível de serviço tem como principal característica o fornecimento de recursos de sistemas de computação na forma de VM e recursos de armazenamento para executar uma demanda de processamento para a publicação e compartilhamento dos objetos de uma pesquisa reprodutível. Portanto, para obter uma infraestrutura para reproduzir um experimento, é necessário acessar o provedor de nuvem pública através de uma rede de computadores, analisar o conjunto de ofertas de recursos de computação, com seus respectivos valores, selecionar a oferta que mais atenda a sua demanda, e ao final, implantar as VMs que possuem toda a infraestrutura necessária para reproduzir seus experimentos.

Diante deste contexto, o ReproeScience agrega os componentes que implementam cada uma das quatro tecnologias em sua arquitetura para integrá-los ao seu mecanismo de reprodutibilidade de *workflows* científicos. O objetivo final da abordagem é prover um mecanismo que armazene e compartilhe a infraestrutura usada para a produção dos resultados publicados em um artigo científico, e a partir desse artigo seja possível recuperar tal infraestrutura e reproduzir o experimento por meio da infraestrutura de um provedor de nuvem, que possui as tecnologia e as VMs necessárias para a reprodução de um experimento.

4.2 Formalização do Objetos de Pesquisa

Sob a visão desta abordagem, um experimento científico é representado por um *workflow* científico. O *workflow* possui um conjunto de elementos computacionais usados para o desenvolvimento de uma hipótese, produção dos resultados e, consequentemente, para a construção de uma conclusão. A hipótese é concebida a partir da definição de uma metodologia de trabalho descrevendo os passos necessários para a execução do experimento, a sequência em que esses passos devem ser executados, dados de entrada e parâmetros de configurações. Portanto, os resultados de um experimento são consequência da combinação deste conjunto de elementos executados sob um determinado número de ensaios para se obter um resultado desejado.

Os elementos que formam um experimento científico apoiado por computador são compostos, em geral, por um conjunto de ferramentas de *hardware*, *software*, documentação (digital e física), parâmetros de configurações e anotações. O resultado e conclusões obtidos com a execução desse experimento são relatados pelos autores do experimento em artigos científicos que são avaliados por revisores, compartilha-

dos por periódicos especializados e comunicados ao público em geral. A Figura 4.1 apresenta a composição desses elementos sob a ótica do ReproeScience.

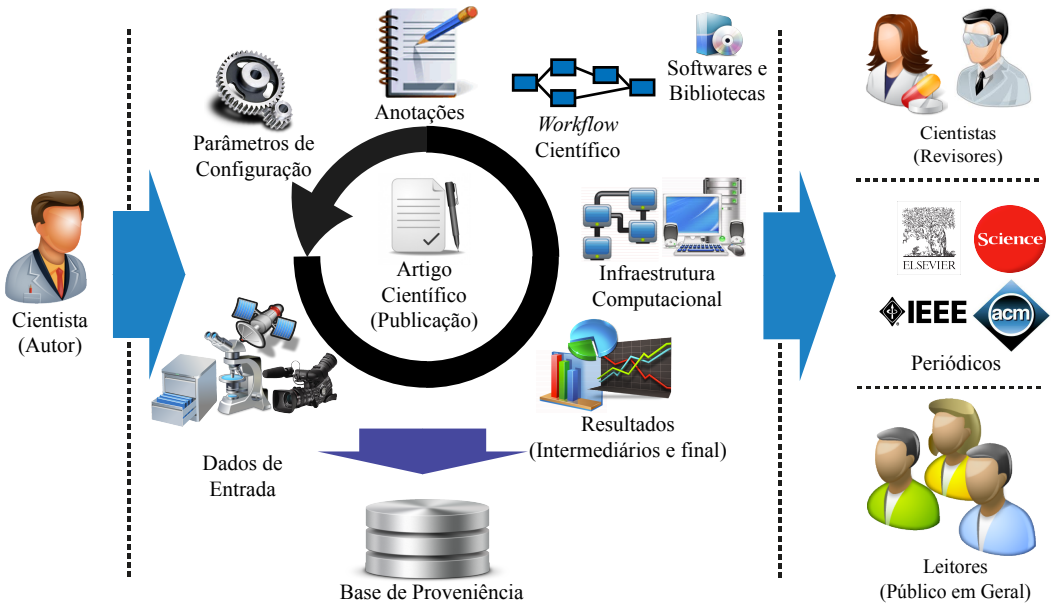


Figura 4.1: Elementos que compõem uma pesquisa reprodutível no ReproeScience.

No contexto do ReproeScience, um cientista (autor) publica os resultados da sua pesquisa em um artigo científico que pode ser apresentado nas versões física (papel) e/ou digital (arquivo *.pdf, *.doc, *.html). Independente da forma de publicação, os resultados são construídos por meio da combinação de um conjunto de elementos que são formados em geral pelo *workflow* científico, que determina cada uma das etapas das atividades do experimento, um conjunto de ferramentas de *software*, que são formadas em geral pelo SGWfC, aplicativos, bibliotecas e *scripts*, um conjunto de parâmetros de configuração de cada uma das atividades que compõem o *workflow*, a infraestrutura computacional (*hardware*), que representa o ambiente em que os experimentos foram conduzidos, o conjunto de dados de entrada e saída e as anotações com as observações tomadas em cada passo.

Todo este conjunto de elementos devem ser registrados em uma base de dados para que seja possível rastrear todos os estados e comportamentos da execução do experimento para a derivação e produção do resultado final. Com este conjunto de informações, é possível para revisores, editoras e público em geral, ter acesso ao protocolo científico e ao conjunto de elementos usados para a execução do experimento, bem como ao conjunto de ensaios realizados pelo cientista. A partir disto, é possível entender como os resultados foram alcançados e, consequentemente, fazer a devida validação e verificação do método como um todo. Essa base de dados registra os metadados e a proveniência dos dados, e dá embasamento e suporte para que novas pesquisas sejam realizadas a partir dessa publicação.

A formulação do problema da reprodução adotada foi desenvolvida sob o contexto apresentado na Figura 4.1, a partir das definições e restrições obtidas nos Capítulos 2 e 3, assim como a partir dos trabalhos levantados no estado da arte no Capítulo 4. O ReproeScience é aplicado a experimentos da *e-Science* apoiados pela tecnologia de *workflows* científicos. O ReproeScience monitora o *workflow* para capturar todos os elementos usados e produzidos durante a sua execução. Ele identifica os estados e comportamentos desses elementos e armazena as suas informações em uma base de dados. O resultado desse monitoramento é a formação de uma base de dados contendo o conjunto de metadados e dados de proveniência que caracterizam a trilha de execução do *workflow* para a produção do resultado de um experimento.

Os elementos do *workflow* científico, em termos de proveniência, envolvem o conjunto de agentes, atividades (processos) e entidades (artefatos) do W3C PROV. Essa definição foi tomada para permitir a representação desses elementos conforme os tipos de dados e relacionamentos definidos pelo modelo de dados W3C PROV-DM. Portanto, as trilhas de execução do *workflow* científico monitorado pelo ReproeScience foram construídas com base nesse modelo de dados.

A formulação do contexto foi orientada pelas duas definições dadas por Freire et al. [48] sobre o conceito de reprodução na computação, apresentadas na Seção 2.1. A definição dos laboratórios leva em consideração que L corresponde à infraestrutura de computação de origem, usada na execução de um *workflow* w relacionado ao experimento e. Trata-se da infraestrutura computacional usada para a produção dos resultados que serão publicados e compartilhados.

Os ambientes de produção podem ser representados por uma estação de trabalho (computador *desktop* ou portátil), *cluster*, provedor de computação em nuvem ou uma grade computacional, onde o *workflow* foi originalmente executado. O laboratório de destino L' representa a infraestrutura na qual esse mesmo *workflow* será posteriormente implantado e reproduzido. Nesse caso, foi adotada a definição de que o laboratório L' é representado no ReproeScience pela infraestrutura de um provedor de nuvem.

Os experimentos apoiados por computador devem levar em consideração as características de *hardware* e *softwares* básico e aplicativos usados na execução. Essas características foram representadas por Freire et al. [48] pela variável s. No ReproeScience, essas características são denominadas como ambiente operacional e são representadas através de um conjunto de recursos de computadores. Nesse caso, esses recursos são organizados de uma forma hierárquica, em diferentes camadas, e se relacionam por meio de uma relação de aninhamento. O recurso de uma camada inferior está contido dentro do recurso da camada superior na hierarquia de recursos.

A primeira camada é representada pelo *hardware* $H = \{h_1, h_2, ..., h_N\}$, que são os sistemas de computação usados para a execução do *workflow*. A segunda camada é

formada pelo sistema operacional S executado sob o *hardware* h_N. A terceira camada possui o SGWfC/*workflow* W adotado para gerenciar o experimento. A quarta camada possui as atividades $A = \{a_1, a_2, ..., a_M\}$. As atividades são associadas a um programa P e seu conjunto de bibliotecas responsáveis por executar uma ação. A quinta camada possui o conjunto de parâmetros de configuração usados por cada um dos programas $C\{c_1, c_2, ..., c_L\}$ e o conjunto de dados $D = \{d_1, d_2, ..., d_Z\}$. Portanto, o *hardware*/sistema operacional s definido por Freire [48] na Seção 2.1 está representado neste trabalho como o conjunto de elementos $\{H, S, W, A, P, C, D\}$.

Um experimento científico e é consolidado em um tempo t. Essa consolidação determina que no momento t, os resultados gerados pelo experimento e, a partir de um conjunto de entrada d, são aqueles que serão publicados e compartilhados para o público em geral. A partir disto, toda e qualquer reprodução de um experimento será realizada em um momento t' posterior a t. Os dados d utilizados para a produção dos resultados de e devem ser disponibilizados juntos a publicação, ou anexados ao experimento ou através de referência a uma URL. Esses dados d são os dados de entrada do experimento usados pelo primeiro processo executado por um *workflow* científico. Esses dados podem ser substituídos por outros dados de entrada d', desde que respeitem uma estrutura ou padrão, conforme d.

Além disso, um experimento pode produzir um conjunto de dados intermediários que irão alimentar os demais processos do *workflow* científico. Esses dados auxiliam a análise de cada uma das etapas de processamento do experimento, e serão produzidos a medida que o *workflow* for reproduzido. A produção desses dados é consequência dos dados de entrada, ou seja, é dependente do conjunto de dados usados para alimentar o *workflow* científico. Para a abordagem ReproeScience esses dados intermediários são essenciais, pois eles são usados no processo de validação da reprodutibilidade, que será melhor explanado no capítulo 5.

Conforme apresentado nesta Seção, cada um desses elementos são armazenados em uma base de dados. Essa base de dados define a organização desses elementos conforme as definições desta abordagem a partir do modelo de dados de proveniência consolidado na literatura. A representação dos elementos apresentados nesta Seção foram adaptadas as características definidas pelo modelos de dados PROV-DM, e o resultado dessa representação será mostrado e discutido na Seção 4.4.

4.3 Arquitetura do Sistema

O ReproeScience faz a interação com três componentes externos: SGWfC, Monitor de Máquina Virtual (*hypervisor*) e o Provedor de Computação em Nuvem. Internamente, ele é composto por 5 módulos internos, um deles responsável pelo gerenciamento da Base de Proveniência para armazenar as informações da infraestrutura,

estados e comportamento da execução do *workflow*. Este conjunto de tecnologias é utilizado para automatizar o processo de captura e armazenamento das informações de proveniência, auxiliar na criação de pacotes portáveis dos sistemas de computadores, por meio de VMs, autenticar e implantar a infraestrutura e experimento na nuvem, compartilhar os pacotes de VMs em um provedor de nuvem, validar e verificar a reprodução, e ainda, compilar artigos que fazem referência a objetos de pesquisa do ReproeScience. O objetivo é entregar um ambiente preparado para a reprodução do *workflow* científico relacionado a uma publicação científica utilizando os recursos da nuvem pública.

A arquitetura interna do ReproeScience, juntamente com a sua interação com os ambientes operacionais, é mostrada na Figura 4.2. Ele é representado pela caixa no tom de cinza claro, e possui os seguntes compenentes internos: Monitor, Gerenciador de Proveniência, Gerenciador de Reprodução, Gerenciador de Pacotes e Aplicação *Web*. A interação entre os componente internos são representados pelas linhas tracejadas e a ordem do fluxo da informação pelas setas direcionais. A interação com os componentes externos são representados pelas setas com linhas sólidas. Os componentes externos, representados pelos ambientes monitorado e de reprodução, são representados por caixas brancas tracejadas. Existe ainda o artigo científico reprodutível que faz referência ao servidor de aplicação para consultar objetos de pesquisa e comandar a reprodução da pesquisa.

Os ambientes externos ao componente são representados pelos sistemas de computadores e unidades de armazenamento. O sistema de computador na nuvem é representado por uma máquina virtual, já no ambiente de origem pode ser representado tanto por uma infraestrutura de computação física como uma virtual.

O ReproeScience fica encapsulado em dois ambientes computacionais diferentes. Em um primeiro momento, ele fica implantado na estação de trabalho do cientista capturando informações e preparando o ambiente para a reprodução. Em seguida, ele é implantado dentro de VMs na infraestrutura dos provedores de computação em nuvem. O ReproeScience pode ser invocado por alguma tecnologia de gerenciamento de *workflow* científico, por exemplo, por meio dos SGWfCs SciCumulus [32], Taverna [91], Vistrails [92] etc.

O módulo monitor possui três componentes internos: (1) processos, (2) arquivos e (3) computador. O componente de processos de sistema operacional é responsável pelo monitoramento dos processos criados e invocados durante a execução do *workflow*. Ele captura todas as informações sobre a identificação, momento de criação/invocação do processo, parâmetros associados etc. O componente de arquivos monitora o sistema operacional para identificar todos os acessos e geração de arquivos durante a execução. São identificados os nomes dos arquivos, momento em que eles foram manipulados e a operação realizada com eles. O componente de

computação consulta o sistema de computador para verificar as características de *hardware* presentes na infraestrutura (frequência e número de processadores, capacidade de memória, capacidade de armazenamento em massa, vazão da rede). Cada um desses monitores são implantados em um sistema de computador distinto na infraestrutura de computação do experimento.

O módulo de gerenciador de proveniência é formado por três componentes: (1) base de proveniência (2) gerador de documentos XML W3C PROV e (3) mecanismo de consulta. A base de proveniência armazena os dados, metadados e dados de proveniência do experimento. Armazena também as referências para os conjuntos de arquivos usados como entradas de dados, dados intermediários/derivados e resultados finais. Optou-se pelo armazenamento dos arquivos no próprio sistema de arquivos, guardando na base de proveniência apenas as referências devido ao tamanho dos arquivos de dados. O gerador de documentos extrai as informações da base de proveniência para construir um arquivo XML com todos os estados e comportamentos capturados pelos monitores durante a execução do *workflow*. Esse documento é usado para a construção das referências de um artigo científico para os objetos de pesquisa usados no experimento. O mecanismo de consulta é composto por métodos usados para recuperar os objetos de pesquisa armazenados na base de proveniência. Ele recebe a identificação e retorna o objeto de pesquisa associado.

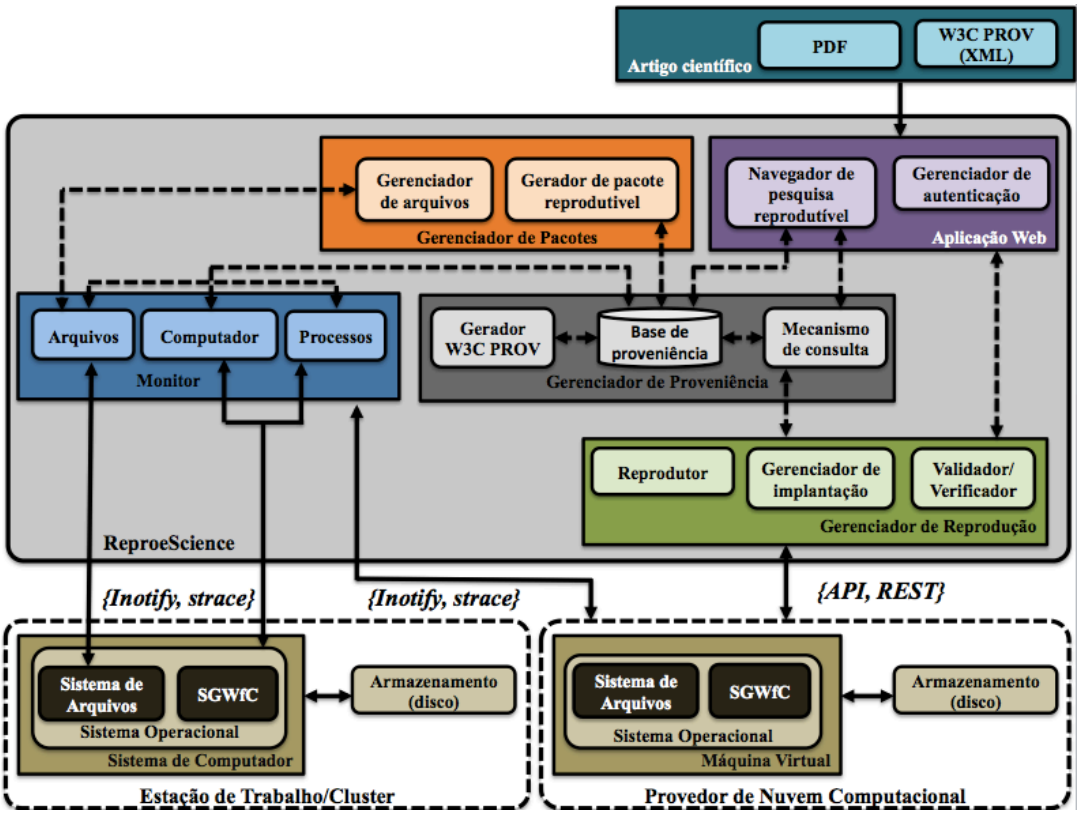


Figura 4.2: Arquitetura interna do ReproeScience.

O módulo gerenciador de pacotes faz o gerenciamento dos pacotes de ambiente, dados e *softwares*. Ele possui dois componentes internos: (1) gerenciador de arquivos e (2) gerador de pacote reprodutível. O gerenciador de arquivos usa o diretório de trabalho do *workflow* relacionado com o experimento em execução e armazena todos os dados de proveniência sob o uso e geração dos arquivos. Após o término da execução, todos os arquivos são comprimidos em um pacote para que seja feita a devida transferência dos objetos de pesquisa para a infraestrutura da nuvem. Em muitos casos, o tamanho do pacote com os arquivos pode ter um grande volume de dados, no entanto, é necessário fazer a transferência para fins de compartilhamento dos elementos relacionados com a pesquisa reprodutível publicada.

O servidor de aplicação *web* do ReproeScience aguarda as requisições para a consulta aos objetos de pesquisa usados na produção dos resultados ou para a reprodução da pesquisa. As consultas recuperam os objetos de pesquisa citados em um documento PDF (compilado com o ReproeScience) e os apresenta em um *web browser*, formando uma documentação da pesquisa reprodutível. As requisições de reprodução aguardam as requisições para a implantação e reprodução do experimento na conta de usuário interessado na reprodução. Esse tipo de requisição deve ser acompanhada das chaves de acesso ao provedor de nuvem. Essas requisições são atendidas pelo módulo de aplicação *web*. Ele é formando por dois componentes: (1) gerenciador de autenticação e (2) navegador de pesquisa reprodutível. O gerenciador de autenticação recebe as credenciais dos usuários que desejam reproduzir o experimento na nuvem validando no provedor. As credenciais são usadas para montar as VMs e os discos na conta do requisitante para que a partir desse ponto ele possa reproduzir o experimento. O navegador é responsável pelas atividades relacionadas à consulta e apresentação dos objetos e pesquisa e, ainda, permite comandar a reprodução.

O módulo gerenciador de reprodução é formado por três elementos internos: (1) gerenciador de implantação, (2) reprodutor e (2) validador/verificador. O gerenciador de implantação faz a preparação da infraestrutura da pesquisa reprodutível na nuvem. Primeiramente, ele acessa o provedor de nuvem por meio das credenciais do cientista requisitando a alocação de espaço nos gerenciadores de armazenamento da nuvem e a instanciação (criação) de uma VM. Em seguida, ele transfere a base de dados, programas (inclusive o SGWfC), bibliotecas e o pacote de pesquisa reprodutível para a VM recém criada na nuvem. Ao final, o gerenciador de implantação extrai o pacote e implanta os programas, bibliotecas e base de dados. O reprodutor de pesquisa pode ser invocado pelo servidor de aplicação, após a infraestrutura do experimento ser totalmente implantada. Ao requisitado, ele invoca o gerenciador de implantação para a criação da infraestrutura da pesquisa na conta do requisitante, que será rotulado leitor ou revisor e, em seguida, apresenta os mecanismos

interativos para a execução do experimento. O validador/verificador permite que um experimento reproduzido possa ser comparado com o experimento originalmente publicado para validar a execução do novo ensaio e para verificar se a pesquisa foi corretamente reproduzida. As comunicações com a nuvem são feitas por meio de APIs ou protocolos REST fornecidas e orientadas pelos provedores de nuvem.

Como a abordagem foi desenvolvida para trabalhar em ambientes de computação paralela e distribuída, no momento da coleta de dados de proveniência, são criadas diversas instâncias do ReproeScience, uma para cada computador usado na execução do *workflow* científico. Pelo fato do componente trabalhar em ambientes remotos ele segue uma arquitetura cliente-servidor. As instâncias clientes do ReproeScience monitoram a execução das atividades de cada nó dos computadores remotos e envia as informações do monitoramento para a instância servidora, que é responsável por armazenar os dados de proveniência na base de dados. A instância servidora do ReproeScience fica hospedada no computador que gerencia o *workflow*, ou seja, fica no nó que possui o SGWfC instalado. As características de coleta de proveniência do ReproeScience tanto em ambientes sequenciais quanto em ambientes paralelo e distribuído são apresentadas na Figura 4.3.

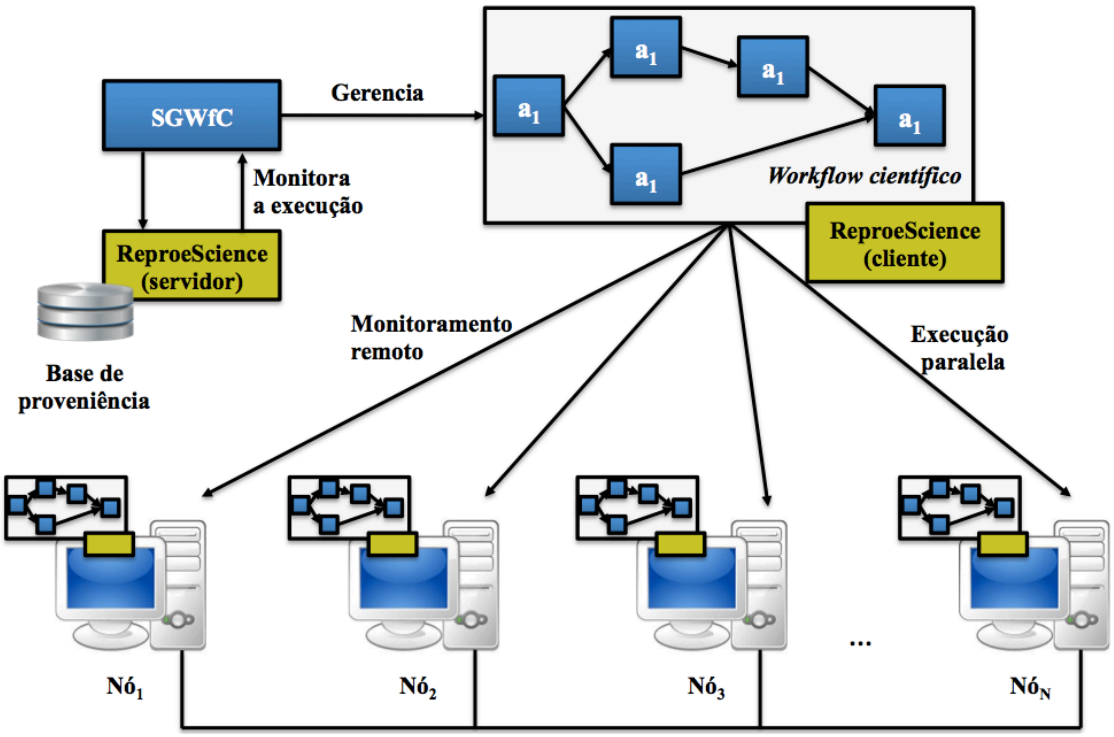


Figura 4.3: Estrutura de execução distribuída do ReproeScience.

Conforme visto nesta Seção, o ReproeScience utiliza um conjunto de tecnologias independentes, tais como SGWfC, MMVs, APIs e bibliotecas de fornecedores de nuvem. Diante dessa diversidade de tecnologias é necessário fazer toda uma preparação do ambiente antes da execução do *workflow* científico. Essa preparação

envolve a instalação do ReproeScience, limpeza de elementos redundantes no sistema de computador que será empacotado (selecionando aplicativos e arquivos que não são usados no experimento), obtenção da chave de acesso com servidores de nuvem, escolha de *hypervisor* na estação de trabalho, dentre outros. A Seção 4.3.1 apresenta os requisitos operacionais e todos os procedimentos necessários para preparar o ambiente para a execução do componente.

4.3.1 Requisitos Operacionais

A abordagem trabalha com ambientes Linux com o *kernel* a partir da versão 2.6.13 (ou superior). O ReproeScience vem acompanhado de um banco de dados armazenado em um SGBD Postgre SQL 9.3 (ou superior). Além disso, é preciso ter suporte para os interpretadores Python 2.7 (ou superior) e Java Standard Edition versão 8.

Antes de iniciar a execução do *workflow* é necessário realizar a configuração do ambiente, preparando as condições essenciais para a execução do monitoramento. O ReproeScience faz o monitoramento dos estados e comportamentos dos arquivos e processos usados e criados durante a execução do *workflow*. Para que o monitor de arquivos possa funcionar corretamente é necessário a existência de um diretório raiz para o *workflow* que será monitorado. Esse diretório deve armazenar todos os arquivos e subdiretórios usados e gerados durante a execução e será a referência onde o *workflow* vai trabalhar efetivamente. A Figura 4.4 apresenta um esquema básico do diretório. Ele deve conter os subdiretórios de atividades, um subdiretório com os dados de entrada, um subdiretório "*templates*", contendo as variáveis de ambiente e diretivas para a instalação dos programas e bibliotecas contidas nos computadores, e o diretório *output* com os arquivos usados e gerados pelas atividades do *workflow*.

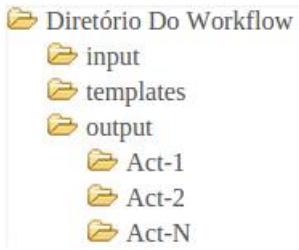


Figura 4.4: Modelo de diretório para o armazenamento dos objetos de pesquisa monitorados durante a execução do *workflow*.

É preciso armazenar o comando que será invocado pelo sistema operacional para a execução do *workflow*. Esse comando será chamado toda vez que for solicitada a execução ou reprodução do experimento. É necessário ainda criar um arquivo XML para a descrição das etapas do *workflow* para que o ReproeScience consiga armazenar os objetos de pesquisa relacionados a cada atividade. Nesse arquivo

também devem constar as informações da conexão do banco de dados e a descrição do nome de cada atividade do *workflow* organizadas na sequência de execução. A seguir, é mostrado um exemplo de arquivo XML com a definição necessária para o ReproeScience configurar o monitoramento. Esse arquivo deve ficar no diretório raiz do *workflow*, definido na Figura 4.4.

```
<workflow>
        <database username="username"
           password="password" port="5432"
           server="localhost"/>
        <conceptualWorkflow description="Scievol Workflow">
                <activity tag="Mafft" />
                <activity tag="ReadSeq" />
                <activity tag="RaXml" />
                <activity tag="Codeml" />
                <activity tag="Analyzer" />
                <activity tag="Compiler" />
        </conceptualWorkflow>
        <executionWorkflow expdir="output/" />
</workflow>
```

A abordagem permite que a infraestrutura do experimento seja empacotada em um conjunto de VMs que representa cada um dos sistemas de computadores usados na execução. Por tal motivo é necessário que o cientista tenha um MMV instalado no seu ambiente. Conforme abordado na Seção 4.3, o ReproeScience trabalha com diferentes tecnologias de MMV e, desta forma, antes de utilizar o componente é necessário que ele selecione o MMV que deseja utilizar e o implante nas estações de trabalho que irão executar cada uma das atividades do *workflow* científico sob o suporte do ReproeScience.

Para o componente ter acesso ao provedor de nuvem, para enviar e publicar as VMs, é necessário que o cientista obtenha as chaves de acesso aos provedores que ele deseja implantar a infraestrutura do seu experimento. Desta forma, é preciso acessar os provedores de nuvem e efetuar um cadastro para obter os dados de autenticação com a nuvem. Além disso, alguns provedores de nuvem determinam que seja implantado um arquivo com uma permissão de acesso na estação de trabalho do cientista. O componente ReproeScience solicita o cadastramento dessas informações antes de executar o monitoramento do *workflow* científico. Portanto, após resolver todos os requisitos de preparação do ambiente, o componente estará pronto para preparar o ambiente do experimento científico para prover os mecanismos para a reprodutibilidade do experimento com a abordagem ReproeScience.

4.4 Modelo de Dados

Os metadados e dados de proveniência são gerenciados pelo módulo Gerenciador de Proveniência apresentado na Figura 4.2. Toda a proveniência está armazenada

em uma base de dados relacional orientada pelo modelo de dados apresentados na Figura 4.5. O núcleo central do modelo é o ***Workflow***, que possui informações sobre o caminho no qual o *workflow* está armazenado dentro do diretório do SGWfC (representado pela entidade ***SGWfC***), e o tempo de inicialização e término de sua execução. Um *workflow* pode ser derivado de um *workflow* já executado (versão), ou seja, pode se tratar de um *workflow* reproduzido que foi modificado e agora traz consigo suas próprias características metodológicas. Esta definição é importante, pois mantém a história de derivação de *workflows* a partir de outros.

Um *workflow* pode ser projetado por um ou mais autores (***Author***), e cada autor pode ter contas de acesso (***Account***) em um ou mais provedores de nuvem (***CloudProvider***). O autor precisa ter suas credenciais previamente cadastradas com a chave de acesso/login e senha para acessar os recursos ofertados pelos provedores (***CloudResources***). O provedor de nuvem é identificado e acessado pelo componente através de uma URL (*Uniform Resource Locator*). Esta URL é o meio pelo qual o fornecimento de recursos pela nuvem ocorre e, consequentemente, é a forma em que tais recursos são obtidos pelo ReproeScience.

O *workflow* é composto por diversas atividades (***Activity***), executadas em um conjunto de ensaios *Test* que identificam em que ensaio do *workflow* um conjunto de resultados foi produzido. Uma atividade possui os atributos descrição, estado de execução e o comando que faz a chamada da execução do *script* ou *software* associado a essa atividade, assim como, o tempo de início e fim da execução. Como um *workflow* pode ser executado em vários ensaios, de forma a testar vários parâmetros e verificar a estabilidade dos resultados gerados por ele, cada uma das atividades possui um número da execução que é responsável por identificar em qual ensaio do *workflow* essa atividade foi executada.

Uma atividade pode consumir um conjunto de elementos, tais como arquivos e parâmetros de configuração, bem como pode produzir um outro conjunto específico desses elementos. Cada um desses elementos são representados pelas entidades (***Entity***). Em geral, uma entidade possui as características de descrição e título, porém, podem assumir diferentes formas com seus próprios atributos. Por tal motivo, uma entidade é definida como uma generalização para as relações arquivos de dados (entidade ***File***), parâmetros de configuração (entidade ***Parameters***) e sistemas de computadores (***ComputerSystem***). As atividades são ligadas com as entidades através de uma relação (***ColActivityEntity***), que identifica uma derivação de **uso**, onde uma atividade utiliza uma entidade, ou de **geração**, que determina que a execução de uma atividade produz uma nova entidade. Ambas as especificações de derivação são armazenadas na relação ***Derivation***.

Uma relação entidade pode estar relacionada com ela mesma através de um auto-relacionamento denominado geração, que determina que uma nova entidade foi

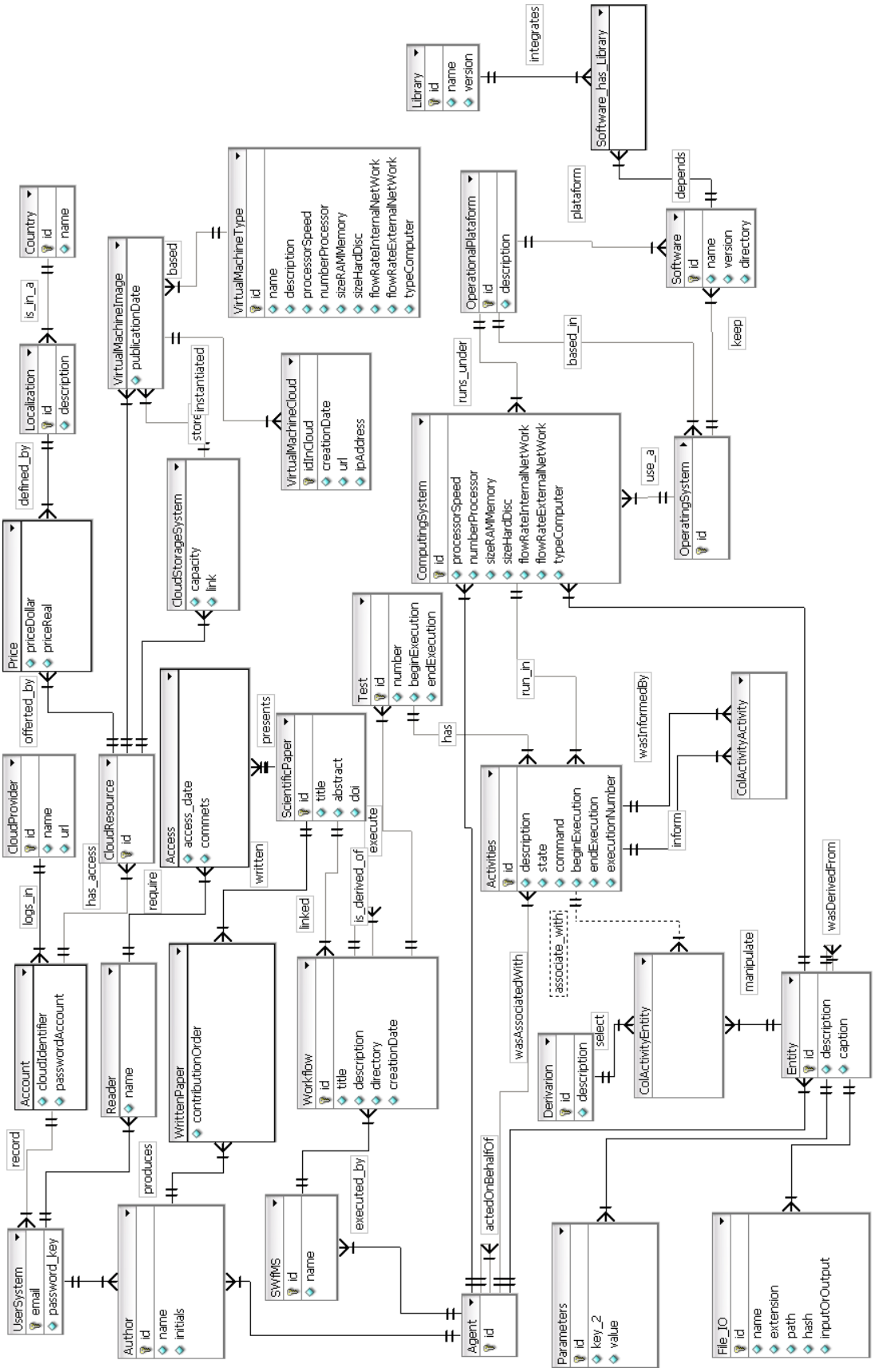


Figura 4.5: Modelo de Dados da base de proveniência do ReproeScience.

criada baseada nas características de uma outra. Da mesma forma, uma atividade pode estar ligada a outra atividade em um auto-relacionamento de comunicação entre as atividades, descrevendo que uma atividade posterior no fluxo de trabalho é comunicada sobre a existência de uma nova entidade, que foi gerada pela atividade anterior no fluxo. Essa comunicação informa que a atividade posterior pode consumir (usar) a entidade gerada pela atividade anterior. Este auto-relacionamento deixa explícita a relação de dependência entre as atividades do *workflow* científico.

Uma atividade também pode ser atribuída a um agente ***Agent***, o qual é uma generalização que pode assumir diferentes formas especializadas, ele pode ser o autor responsável por projetar a atividade, pode ser um *software* (i.e. SGWfC), e até mesmo uma organização, como uma Universidade ou Centro de Pesquisa. Os agentes podem estar associados com outros agentes em um relacionamento onde um agente atribui autoridade e responsabilidade para outro agente na realização de uma ou mais atividades, desempenhando um papel de representante. Por exemplo, é comum um SGWfC executar as atividades de um *workflow* representando o autor do experimento, ou um autor estar representando uma instituição de pesquisa e assim por diante. Um agente se relaciona ainda com uma entidade, onde uma dada entidade é atribuída a um agente dando direitos de propriedade para o mesmo.

Cada atividade pode ser executada em um ou mais computadores. A definição desses computadores utilizados na execução do *workflow* é de suma importância, pois suas características irão orientar a obtenção dos recursos para a reprodução na nuvem. Tais computadores, representados pela entidade ***ComputingSystem***, são caracterizados pelos atributos de quantidade e velocidade de processador, capacidade de memória e armazenamento em massa, e vazão das redes internas e externas. São identificadas também a plataforma operacional na qual o sistema opera (32 ou 64 bits atualmente), o sistema operacional instalado e os *softwares*/bibliotecas que foram utilizados pelo *workflow*, os três últimos representados, respectivamente, por ***OperationalPlataform***, ***OperatingSystem*** e ***Software***.

A abordagem reproduz os sistemas de computadores com os recursos oferecidos na infraestrutura da nuvem. Portanto, para possibilitar tal reprodução, o ReproeScience cria as VMs baseadas nas características operacionais de cada um dos sistemas de computadores usados na execução do experimento original. Para reproduzir o sistema de computador na nuvem, a VM gerada deve ser compatível com o provedor de nuvem que o cientista deseja publicar o experimento, portanto, essa VM deve ser associada a uma Imagem de Máquina Virtual (VMI) do próprio provedor de nuvem. Essa VMI é uma estrutura usada para a criação da VM que representa o sistema de computador. Portanto, antes da VMI ser criada ou enviada para a nuvem é feita uma adaptação dos sistemas de computadores gerados a partir do *workflow* original em relação aos recursos fornecidos pelo provedor de nuvem.

A VMI é associada a um tipo de recurso fornecido pelo provedor de nuvem e, em seguida, é mantida no sistema de armazenamento da nuvem (***CloudStorageSystem***), vinculada na conta do cientista (relação ***Account***). Desta forma, essa VMI fica associada com a conta do cientista no fornecedor de computação em nuvem selecionado para publicação e compartilhamento do *workflow*. A partir desse momento, os sistemas de computadores, o SWfMS, o *workflow* científico e os demais insumos estão prontos para serem obtidos e implantados.

Os interessados na reprodução do experimento podem selecionar a VMI compartilhada no provedor de nuvem, e implantar os sistemas de computadores usados na execução do experimento sob a forma de uma VM, que é identificada pela entidade ***VirtualMachine***. Quando a VMI é criada, ela é associada a um determinado tipo de recurso do fornecedor de nuvem, neste caso representado pela entidade ***TypeVirtualMachine***. Portanto, assim como todo o tipo de recurso nos fornecedores de nuvem pública, a VM fica vinculada a uma precificação (***Price***). O valor do recurso é dependente do país que ele foi alocado e muitas vezes também é baseado na localização (estado, província, cidade) do recurso dentro do pais selecionado. A localização é representada pela relação ***Localization***.

Conforme mencionado anteriormente, o armazenamento da proveniência é feita em uma base de dados relacional, porém, para atender aos padrões de representação da proveniência foi adotada uma estrutura de arquivos XML (***Extensible Markup Language***), seguindo as orientações do PROV-DM e do PROV-XML [93]. Essa representação é mostrada na Figura 4.6, onde é possível verificar cada um dos tipos de dados e relacionamentos do W3C Prov. A estrutura XML é formada com base nas informações contidas na base de dados relacional que foi construída conforme o modelo de dados da Figura 4.5. Os vértices no grafo representam os tipos de dados do Prov-DM com as entidades (amarelo), as atividades (azul) e os agentes (cinza). Cada um desses vértices possui um conjunto de atributos, que também são representados neste modelo como arestas do grafo.

A representação em grafo é de suma importância, pois, a etapa de avaliação da confiabilidade da reprodutibilidade proposta nesta tese, verifica se o conteúdo de cada um dos vértices do grafo e, consequentemente, dos seus respectivos atributos, são equivalentes. Esta etapa envolverá a comparação dos vértices e arestas do *workflow* original com os *workflows* reproduzidos. E para tornar essa análise mais conveniente, os atributos de cada um dos agentes, atividades e entidades do *workflow* devem ser transformados em vértices no grafo que representa esse *workflow*, de forma que esse conjunto de informações sejam comparadas neste nível de detalhe.

A Figura 4.6 mostra as relações ***File***, ***Parameter***, ***VirtualMachineCloud*** e ***ComputerSystem*** associadas com uma atividade (***Activity***) através de uma aresta ***Used*** (marrom). A entidade ***File*** possui uma associação por geração - Was-

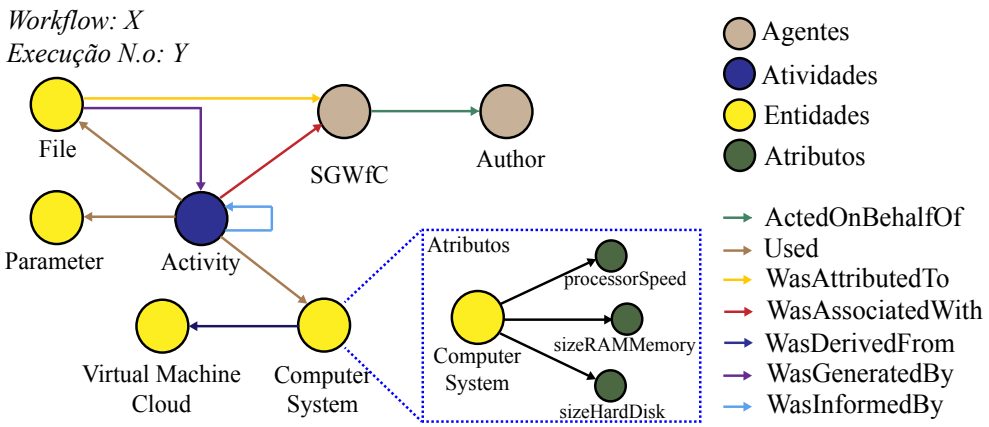


Figura 4.6: Modelo de Dados da base de proveniência do ReproeScience.

GeneratedBy (roxo) - com uma atividade, e uma atividade está relacionada com outra atividade através de uma associação WasInformedBy (azul claro). A entidade ***ComputerSystem*** foi utilizada como base para a produção da entidade ***VirtualMachineCloud*** via associação de derivação - WasDerivedFrom (azul escuro). A entidade ***ComputerSystem*** tem alguns de seus atributos apresentados na Figura, sendo eles, o *processorSpeed*, *sizeRAMMemory* e *sizeHardDisk*.

Os agentes apresentados no grafo são representados pelo autor (***Author***) e pelo ***SGWfC***, o último é responsável por modelar e executar o *workflow*. O agente ***SGWfC*** tem um entidade ***File*** atribuída a ele através da associação WasAttributedTo (amarelo). O agente é ligado com atividades através da associação WasAssociatedWith (vermelho), e se relaciona com outros agentes através da delegação com o ActedOnBehalfOf (verde). É importante verificar que as arestas que associam os vértices são unidirecionais conforme as restrições do W3C-CONSTRAINT [94].

A representação com o modelo de grafos é extraída para um arquivo XML que é responsável por toda a descrição da proveniência dos experimentos executados com o ReproeScience. As informações contidas nesse arquivo serão referenciadas por um conjunto de *tags* do ReproeScience inseridas no conteúdo de um artigo escrito em Latex. A reprodução do conteúdo do artigo apoiado pela abordagem será orientada pelas informações contidas nesse documento XML. É importante ressaltar que, além das informações de proveniência do componente, o cientista tem acesso a dados de proveniência complementares, contidas na base de dados do SGWfC responsável por gerenciar o *workflow*, e podem utilizar ambas as bases de dados para complementar as informações de proveniência do seu experimento. E, uma vez com as informações armazenadas na base de proveniência, é possível recuperar a infraestrutura e reproduzir o experimento conforme a sua execução original através de um conjunto de procedimentos que serão mostrados nas próximas seções.

4.5 W3C PROV no ReproeScience

Esta Seção apresenta as notações usadas para a construção dos objetos W3C PROV usados para registrar a proveniência. Trata-se das definições das famílias de documentos PROV-N e PROV-DM adaptados ao contexto de reprodução tratado neste trabalho. São definidos os agentes, atividades e entidades, bem como os relacionamentos e demais informações de proveniência.

As entidades foram denotadas como: $entity(id, [attr_1 = val_1, ..., attr_N = val_N])$, onde o id é um identificador da entidade e os atributos, que são opcionais, são formados pelos pares de atributo-valor $(attr_i, valor_i), i = 1, \ldots, N$. As atividades são escritas no formato $activity(id, st, et, [attr_1 = val_1, ..., attr_N = val_N])$, onde, o id é um identificador para uma atividade, st (*Start Time*) é um atributo opcional que identifica o início da atividade, et (*End Time*) define o final de uma atividade. Além disso, uma atividade é composta por um conjunto de atributos opcionais, formados pelos pares de atributo-valor $(attr_i, valor_i), i = 1, \ldots, N$.

A geração é escrita na forma $wasGeneratedBy(id; e, a, t, attrs)$, vindo e a ser a entidade criada, a a atividade responsável pela criação, t o tempo (ou momento) em que uma entidade foi criada (gerada), e $attrs$ apresenta o conjunto de atributos que complementam as informações de geração. Essa relação pode ser expressa como $Used(id; a, e, t, attrs)$, onde e identifica a entidade que será usada, a identifica a atividade que irá usar a entidade e, t especifica o momento em que a atividade a irá usar a entidade e, e $attrs$ são os atributos adicionais dessa relação. Uma comunicação é escrita como $wasInformedBy(id; a2, a1, attrs)$, com a_2 descrevendo a atividade informada da geração da nova entidade, a_1 sendo responsável por informar a atividade a_2 que ela gerou uma entidade pronta para ser consumida, e $attrs$ sendo a lista de atributos complementares opcionais.

A inicialização é escrita como: $WasStartedBy(id, a_2, e, a_1, t, attrs)$, sendo que a_2 indica o identificador da atividade inicializada, e informa qual foi a entidade gatilho da atividade, a_1 indica qual foi a atividade que gerou a entidade e. Além disso, são informados o momento t em que a atividade foi inicializada, e os pares de atributos-valores representando as informações adicionais. A finalização é apresentada na forma $WasEndedBy(id; a_2, e, a_1, t, attrs)$, alterando-se em relação a inicialização apenas a aplicação de a_2, que aqui indica a atividade finalizada, e que representa a entidade gatilho da finalização. Uma invalidação é escrita como $wasInvalidatedBy(id; e, a, t, attrs)$, sendo que e identifica a entidade que está sendo invalidada, enquanto a identifica a atividade que está invalidando a entidade, t determina em que momento a invalidação está sendo feita e $attr$ identifica a lista de atributos opcionais que podem ser registrados com esse processo de invalidação.

Uma derivação é escrita na forma $wasDerivedFrom(id; e2, e1, a, g2, u1, attrs)$,

onde id é o identificador, e_1 identifica a entidade usada na derivação, e_2 é a entidade que foi derivada a partir de e_1, a representa o identificador da atividade responsável pelo uso ou geração das entidades e_2 e e_1, os parâmetros g_2 e u_1 são parâmetros usados para identificar, respectivamente, a entidade gerada e a entidade usada. O último parâmetro, $attrs$ é formado pela lista de todos os atributos usados para enriquecer as informações sobre a derivação. Os parâmetros a, g_2 e u_1 são opcionais.

A derivação é denotada através da declaração do tipo $prov : Revision$, que é declarado dentro da relação $WasDerivedFrom(id, e_1, e_2, [prov : type =' prov : Revision'])$, que define que uma entidade e_1 foi derivada de uma entidade e_2 através de um processo de revisão. A citação é declarada através do tipo $prov : Quotation$, e deve ser usada da mesma forma que na revisão, ou seja, dentro de uma relação $WasDevivedFrom(id, e_1, e_2, [prov : type =' prov : Quotation'])$. Esse tipo de relação é denotada como $prov : PrimarySource$. A definição para a identificação de uma fonte primária é declarada dentro de uma relação de derivação, utilizando-se a sintaxe $WasDevivedFrom(id, e_1, e_2, [prov : type =' prov : PrimarySource'])$.

Um agente é definido da seguinte forma: $agent(id, [attr_1 = val_1, ..., attr_N = val_N])$, onde o id é um identificador da entidade e $attr$ são formados pelos conjuntos de atributos opcionais. É importante ressaltar que um agente pode ser ainda um tipo particular de entidade ou atividade. A definição dos tipos de agentes podem ser feitas da seguinte forma: $agent(e1, [prov : type =' prov : Person'])$.

Uma associação é escrita da seguinte forma: $wasAttributedTo(id; e, ag, attrs)$, onde id é o identificador da relação de atribuição, e identifica a entidade atribuída ao agente ag e $attr$ é formado pelo conjunto de atributos opcionais. A associação é escrita na forma $wasAssociatedWith(id; a, ag, pl, attrs)$, onde id é o identificador da relação de atribuição, a identifica a atividade na qual o agente ag desempenha um papel, pl define o plano do agente ag no contexto da atividade a e $attr$ e é formado pelo conjunto de atributos opcionais. A definição de um plano deve ser feita através do tipo $prov : type = prov : Plan$ dentro da definição de uma entidade.

A delegação é definida como $actedOnBehalfOf(id; ag2, ag1, a, attrs)$, sendo que o id é o identificador, ag_2 é o agente associado com a atividade a que age em nome do agente responsável por ela, ag_1 identifica o agente que delegou a atividade a para o agente ag_2, e $attr$ é formado pelo conjunto de atributos opcionais.

Uma relação de especialização é escrita como $specializationOf(infra, supra)$, sendo que o parâmetro $infra$ identifica a entidade especialista, que irá receber todos os aspectos das entidades mais genérica. O parâmetro $supra$ identifica a entidade genérica, detentora dos aspectos que serão compartilhadas por todas as entidades especializadas a partir dela. A relação alternativa é escrita na forma $alternateOf(e_1, e_2)$, sendo que e_2 representa uma alternativa de e_1. Essa relação pode significar que e_1 possivelmente evoluiu para e_2 em algum momento no tempo,

tornando-se uma conexão relevante de proveniência entre essas entidades.

A localização pode ser definida da seguinte forma: $entity(e1, [prov : location = "/home/aryhenrique/Document"])$. O valor associado com um atributo $prov : rule$ deve ser um valor Prov-DM. Uma regra é definida como: $prov : role = "author"$.

4.6 Ciclo de Vida

A abordagem trabalha para a concepção de um mecanismo para a publicação e compartilhamento de experimentos científicos, para que seja possível referenciar e reproduzir esse ambiente juntamente com os seus agentes, atividades e entidades, dando acesso a todos os objetos de pesquisa de um experimento. A abordagem exige a definição de cada elemento que forma o experimento segundo um conjunto de padrões de proveniência consolidados, e utiliza-se de três ambientes operacionais bem definidos para construir (estação de trabalho), compartilhar (fornecedores de nuvem) e referenciar (servidor *web*) os elementos que compõem o experimento. Cada uma das etapas e ambientes envolvidos no mecanismo de reprodução está representada na Figura 4.7.

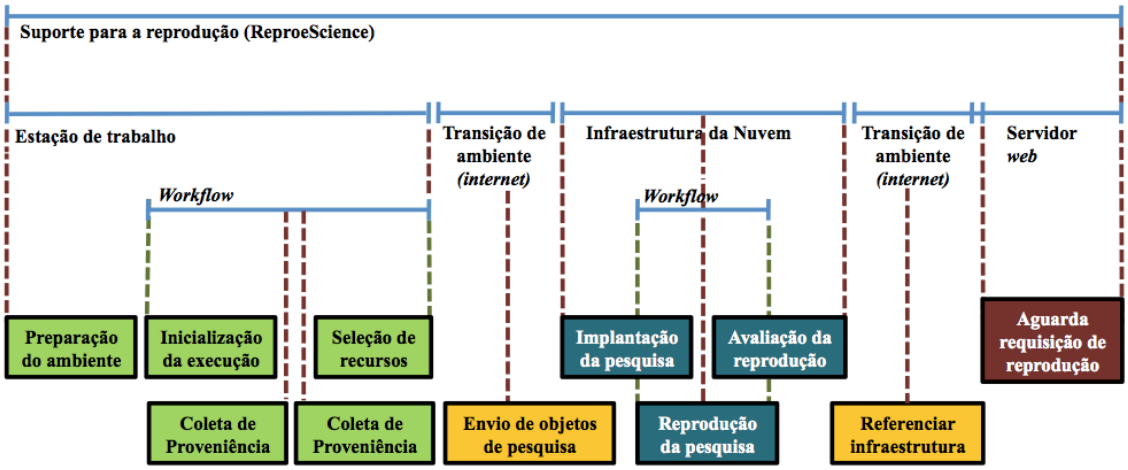


Figura 4.7: Cenário de execução e ciclo de vida do ReproeScience.

A abordagem utiliza-se de três etapas gerais bem distintas para o apoio da reprodução. Estas etapas envolvem a preparação da infraestrutura para a reprodução na estação de trabalho do cientista, o armazenamento, publicação e compartilhamento dessa infraestrutura no provedor de computação em nuvem e a construção da referencia da infraestrutura através de um artigo executável via servidor *web*. Portanto, os mecanismos usados para a reprodução transitam por diferentes ambientes operacionais separados pela *internet*. Primeiramente, envia as VMs da infraestrutura local preparada para o armazenamento, publicação e compartilhamento para a nuvem e, em seguida, o servidor *web* implantado aguarda as requisições de referência

para acionar a reprodução da infraestrutura do experimento na nuvem.

É importante observar que na primeira fase, na estação de trabalho do cientista, seja ela local ou remota, grande parte das etapas de preparação para a reprodutibilidade são executadas juntamente com o *workflow* científico, e que toda essa infraestrutura será implantada no provedor de nuvem como um espelho do experimento. Esse espelho do experimento é publicado e compartilhado na forma de uma VMI. Na segunda fase, essa VMI será usada no provedor de nuvem como um molde no qual outros usuários vão instanciar e implantar as suas VMs, e uma vez implantadas, essas VMs podem reexecutar o *workflow* consolidado e compartilhado pelo cientista, ou vão criar novas variações a partir desse *workflow*. Essas variações são relacionadas a mudanças nos conjuntos de dados de entrada, parâmetros de configuração, ou até mesmo com a inclusão ou exclusão de atividades do *workflow*.

Se esses outros usuários, que estão reproduzindo o *workflow*, decidirem fazer novos testes de parâmetros ou melhorias no experimento, essas modificações não irão afetar o experimento originalmente compartilhado na nuvem, mas irão criar uma nova derivação do *workflow*. A garantia da integridade do experimento original é feita através da associação de um valor de *hash* de uma VMI disponibilizada na nuvem do experimento que irá diferenciar o que foi publicado daquele que foi reproduzido. Esse valor de *hash* ajuda a controlar a segurança, pois garante que os elementos compartilhados foram aqueles usados no experimento, e ainda permite avaliar o conteúdo dos arquivos de dados reproduzidos, comparando-os em relação aos conjuntos de arquivos da execução original.

Cada um dos passos seguidos pela abordagem para a reprodução, desde a preparação do ambiente até a reprodução de um artigo executável são definidos nas próximas seções.

4.6.1 Preparação do Ambiente

O ReproeScience atua em uma solução de *workflow* projetada e consolidada por um cientista, ou seja, o ambiente de execução já deve estar preparado para rodar o experimento. A preparação do ambiente é responsabilidade do cientista, e deve incluir as linguagens java e python, para a execução da abordagem, bem como os componentes que formam o *workflow* e as chamadas aos métodos do ReproeScience para coletar e armazenar as informações necessárias para reproduzir o experimento. O ReproeScience foi projetado para reproduzir experimentos apoiados por computador e, para isso, é necessário que exista pelo menos uma primeira execução monitorada pelo componente, para que o mesmo possa fazer uma cópia do ambiente operacional e, posteriormente, implantá-lo e reproduzi-lo na nuvem.

Após a biblioteca do ReproeScience ser incluída nos códigos de chamada e ge-

renciamento do *workflow*, o cientista deve identificar quais serão os agentes, as atividades e as entidades que compõem o experimento, bem como, quais são as associações entre esses componentes segundo as orientações do W3C PROV. A definição dos componentes é feita por meio da invocação dos métodos do ReproeScience, e esses métodos têm como parâmetros a caracterização dos elementos, tais como, descrição, momento da criação do componente (data e horário), qual o agente ou evento responsável pela definição deste componente, e o local onde esses componentes encontram-se armazenados etc. As características do armazenamento da proveniência segue as orientações do modelo de dados apresentado na Seção 4.4.

As anotações sobre as atividades devem informar o momento em que a sua execução será iniciada. O marco que identifica esse início é representado pela execução do comando de criação do diretório de trabalho dessa atividade, dentro do diretório do *workflow*. Esse marco identifica que a partir desse momento, todos os objetos de pesquisa manipulados durante a execução dessa atividade serão armazenados dentro deste diretório. O arquivo XML apresentado na Seção 4.3.1 demonstra um exemplo de criação dos diretórios de trabalho de um *workflow*. Consequentemente, o cientista deverá informar as atividades no arquivo XML para que ReproeScience identifique qual atividade que um diretório representa.

Após definidas as atividades, o componente associa todos os arquivos lidos e gerados pela atividade corrente durante a execução do *workflow*. Os arquivos são associados a cada um dos diretórios informados pelo cientista no XML. Todos os arquivos lidos e criados durante a execução são associados com a devida atividade responsável pela sua manipulação e são definidos como entidades. Os arquivos lidos geram uma associação *used* com a atividade corrente e os arquivos criados geram uma associação *wasGeneratedBy*. As atividades são associadas ao programas usados para a execução dos processos de manipulação dos dados. Esses programas são definidos como agentes, e automaticamente são associados com a atividade no qual está relacionado por meio do relacionamento *wasAssociatedWith*, assim como com os arquivos lidos e produzidos por meio do relacionamento *wasAttributedTo*.

A completa execução do ReproeScience é finalizada com a implantação da infraestrutura do experimento em um provedor de nuvem. O cientista precisa informar os dados da sua credencial no provedor para que o componente consiga automatizar a configuração, envio e inicialização dos serviços e prover o serviço de publicação, compartilhamento e reprodução da pesquisa reprodutível. Esse processo exige o preenchimento de um arquivo XML, contendo o nome e identificação do provedor de nuvem, assim como o nome e chave de acesso do usuário neste provedor de nuvem associado, conforme mostrado a seguir.

```
<cloud provider="provider name">
    <cloudId></cloudId>
    <cloudKey></cloudKey>
```

```
    <cloudPassword></cloudPassword>
</cloud>
```

Nesta etapa, ainda não ocorre a execução do experimento, apenas as configurações necessárias para o monitoramento do *workflow* pelo ReproeScience. Portanto, após a ambiente estar com todos os requisitos previamente atendidos, contendo as credenciais e arquivos de permissão para que o componente possa integrar com os provedores de nuvem, ele estará pronto para iniciar a sua execução em paralelo com a execução do *workflow* científico no SGWfC.

4.6.2 Inicialização

O processo de inicialização é a primeira etapa de execução do componente. Ela é responsável pela preparação do ambiente de execução do *workflow* científico sob o suporte da abordagem de reprodução. Ele verifica se todos os componentes estão carregados e prontos para a inicialização das atividades do ReproeSceince. Esta etapa envolve as atividades de preparação do mecanismo de monitoramento da execução, a inicialização e preparação da base de proveniência do ReproeScience, a identificação do número de nós de execução e a preparação da estrutura de diretórios no sistema de computador local para receber os arquivos temporários do componente.

A primeira atividade é a preparação do mecanismo de execução, ela faz a identificação de cada um dos sistemas de computadores que irão compor e processar o experimento. Esse mecanismo é utilizado para que a abordagem possa reproduzir experimentos que usam tanto a estratégia sequencial quanto a estratégia paralela na execução das atividades. Muitas arquiteturas do HPC possuem diversos nós de processamento que utilizam a comunicação via rede de computadores, portanto, é necessário identificar quantos (*number of processes*) e quais nós (*IP address*/access port/node rank identifier) estão envolvidos no processamento do *workflow* para que seja possível fazer o monitoramento em cada um deles.

O ReproeScience possui uma base de dados própria que fica implantada no mesmo sistema de computador que faz o gerenciamento do *workflow*. Essa base de dados é identificada pelo endereço de rede (endereço IP - *Internet Protocol*) e pela porta de conexão com o banco de dados. Se o experimento for implementado com uma estratégia paralela, todos os computadores (nós) envolvidos na execução das atividades do *workflow* enviam os dados de proveniência para essa base de dados.

A reprodução exige o acesso a todos os elemento usados na execução do experimento original. Existem elementos, tais como os conjuntos de dados de entrada do experimento, *softwares* e bibliotecas, que possuem um tamanho em *bytes* com grande volume de dados e, muitas vezes, não podem ser armazenados em banco de dados. Para tratar esses casos, o ReproeScience usa a estrutura de diretórios criada

na fase de preparação para armazenar esses elementos, identificando o experimento e a execução na qual o conjunto de elementos foi utilizado.

4.6.3 Monitoramento para Coleta da Proveniência

A etapa de monitoramento está relacionada com o módulo monitor da Figura 4.2. Esse módulo captura as informações dos sistemas de computadores usados pelo *workflow*, processos invocados para a execução dos programas associados com as atividades e o conjunto de arquivos de dados de entrada e saída, ou seja, usados e gerados. O monitoramento ocorre em nível de sistema por meio do monitoramento de arquivos do sistema operacional Linux (*kernel* a partir da versão 2.6) e chamadas de processos. O sistema de monitoramento de arquivos recebe um parâmetro contendo o diretório raiz do SGWfC como o diretório inicial no qual o sistema deverá identificar as modificações no gerenciamento dos arquivos. Ele monitora todos os diretórios recursivamente, tanto os já existentes quanto os que serão posteriormente criados. O monitoramento do sistema de arquivos é baseado nos eventos de acesso, criação e movimentação. O processo de acesso é caracterizado pela leitura de um arquivo. A criação é o resultado da geração de um novo arquivo dentro de um diretório. A movimentação trata a transferência do arquivo de um diretório de origem para um diretório de destino.

Quando uma atividade do *workflow* acessa um arquivo, o monitor identifica que essa atividade está usando essa entidade do tipo arquivo. Quando um diretório é modificado através da criação de um novo arquivo, o monitor identifica que essa ação trata-se de uma geração de uma entidade arquivo por uma atividade. Quando um *workflow* copia um arquivo de um diretório para o outro, o monitor identifica essa ação como um processo de derivação, onde a cópia foi derivada do arquivo original. A Figura 4.8 mostra o exemplo de dois eventos capturados pelo monitor *Strace* referente a abertura (*open*), denominado uso (used) no W3C PROV, e renomeação (*renament*) de arquivos de dados.

```
open("/home/users/ary/scievol/template_mafft/experiment.cmd", O_RDONLY|O_NOFOLLOW) = 3
rename("/home/users/ary/montage/exp/ListFits/2/aJ_asky_990221n1040068.fits", "/home/users/
ary/montage/exp/Projection/3/aJ_asky_990221n1040068.fits") = 0
```

Figura 4.8: Eventos de abertura e renomeação de arquivos realizada pelos monitores

Foram desenvolvidas duas implementações do monitor de arquivos, a primeira utilizando a ferramenta *Strace*[1], que é uma implementação do mecanismo de monitoramento de sistema *Ptrace* e, a segunda denominada Repromonitor, que utiliza um conjunto de bibliotecas nativa do sistema operacional Linux, denominada *inotify*. O *inotify* é um subsistema do *kernel* do Linux que monitora eventos executados

[1]http://linux.die.net/man/1/strace

sob os arquivos do sistema operacional. O Repromonitor aproveita essa capacidade do *inotify* para identificar as operações executadas sob o diretório do *workflow*. A segunda implementação também utilizou mecanismos de interceptação da função de acesso aos arquivos para discriminar o tipo de operação realizada. A Figura 4.9 apresenta as duas estratégias de implementação do monitor de arquivos.

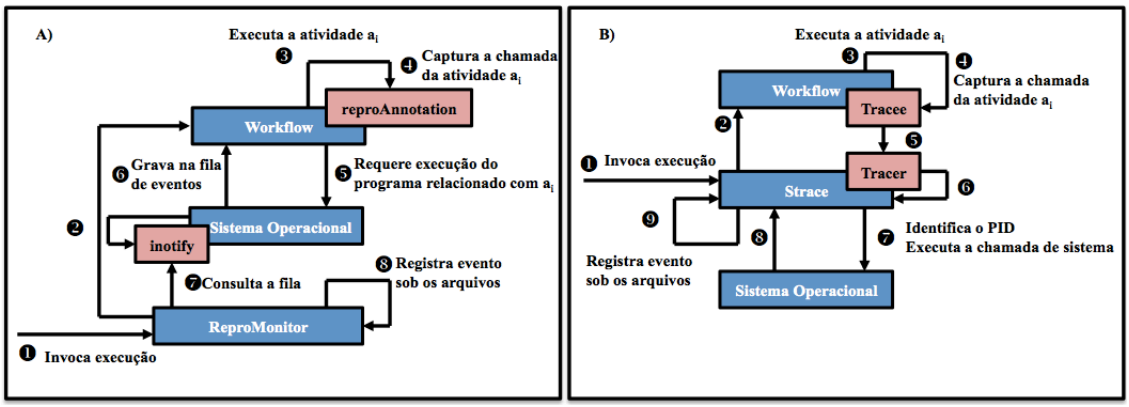


Figura 4.9: Estratégias de monitoramento de arquivos Strace (dir.) e *inotify* (esq.).

O monitor representado à esquerda na Figura 4.9 é invocado (1) para acompanhar a execução do *workflow*. Diante disto, ele solicita a execução do *workflow* (2) e a medida que as atividades são executadas (3) é feita captura dos diretórios de arquivos de dados associados com a atividade em execução etapa (4). Todas as chamadas para o sistema operacional (5) são armazenadas em um repositório consumido pela ferramenta *inotify*. O monitor, aqui denominado Repromonitor (ou Repro), consome as informações obtidas pelo *inotify* sem interferir na execução (7) e registra os eventos sob os arquivos.

Já o monitor representado à direita é invocado (1) da mesma forma para o acompanhamento da execução do *workflow*. Esse monitor é baseado nos componentes *Strace/Ptrace* para fazer a captura de chamadas de sistema. Ele faz a devida chamada para a execução do *workflow* (2) e a medida que as atividades são executadas (3) é feita uma captura do comando de chamada de sistema por um cliente do *Ptrace* denominado *tracee* (4). A chamada de sistema capturada é enviada para o *Strace* (5), por meio do *Tracer* (6), que registra a chamada de sistema capturada, para em seguida, passar o comando para o sistema operacional (7). Esse monitor interfere diretamente na execução do *workflow* inserindo uma sobrecarga para fazer o monitoramento.

Existem três operações básicas sob arquivos que são capturadas pelos monitores: criação, movimentação e as cópias de arquivos. Fazendo uma analogia com o padrão W3C PROV, a criação dos arquivos representa a operação de geração de uma nova entidade por um processo ligado a uma determinada atividade. As demais operações, movimentação e cópia, representam a operação de uso das en-

tidades por uma atividade. O monitor de arquivo constrói uma representação de proveniência associando os processos de computador com as atividades, os arquivos com as entidades e monta o relacionamento de geração e uso entre as entidade e as atividades, traçando o histórico de derivação de um resultado. Durante a execução das atividades é necessário movimentar um arquivo de um local para outro. Essa movimentação acontece na troca de informações de uma atividade antecessora para uma atividade sucessora. Essa operação exige a identificação e interceptação da função de abertura do arquivo do programa de cópia de arquivos.

Os dois monitores desempenham o mesmo papel, ou seja, ambos fazem o monitoramento das operações efetuadas sobre os arquivos durante a execução do *workflow* científico. Entretanto, o *Strace* funciona como um mecanismo de depuração capturando todas as interrupções e chamadas ao sistema operacional, e o *repromonitor* é ativado somente quando ocorre um evento relacionado aos arquivos ligados ao diretório de trabalho do *workflow*.

Ambos os monitores foram implementados para acompanhar a geração e a utilização dos arquivos manipulados nos sistemas de arquivos dos computadores remotos utilizados para a execução de um *workflow*. No caso específico do repromonitor, cada computador remoto recebe uma instância cliente dos monitores, e a medida que o sistema de arquivo sofre modificações, os clientes enviam as informações sobre essas modificações para o monitor servidor. As informações são referentes a denominação do diretório e são comunicadas para todos os outros clientes para que eles tenham conhecimento desse novo diretório, e possam, consequentemente, passar a monitorá-lo. Portanto, o repromonitor utiliza um canal de comunicação exclusivo para informar essas criações de diretórios.

No caso do *Strace* não é necessário efetuar o compartilhamento de informações sobre novos diretórios, pois, como o *Strace* faz a interceptação de chamadas de sistema e interrupções, ele tem conhecimento sobre todos os eventos e informações sobre um determinado arquivo. Para manter a associação dos arquivos com as atividades é necessário instrumentar o *workflow* de forma que sejam criadas pastas relativas a cada atividade e, consequentemente, todos os arquivos vinculados a essa atividade devem ser criados ou movidos para essa pasta.

O sistema de monitoramento de chamadas de processo recebe os comandos necessários para a execução do *workflow* no SGWfC. Ele executa o comando e obtém o número de identificação do processo (*process identificator* - PID). O monitor usa o valor do PID para obter todas as informações sobre as execuções dos programas derivados do processo vinculado a esse PID. São obtidas as informações do nome dos programas, os parâmetros usados na chamada da execução e o diretório de trabalho. O monitor cria uma árvore de dependência que mostra a hierarquia de chamadas dos processos. A medida que o *workflow* é executado todas as informações sobre

os processos são vinculadas à árvore de execução, tornando possível identificar os programas e bibliotecas necessários para a posterior reprodução.

O algoritmo 4.6.3 apresenta os passos usados na implementação do *Strace* para identificar os *softwares* e bibliotecas invocados durante a execução do *workflow*. Inicialmente, são declaradas 2 variáveis responsáveis por listar: (1) os arquivos abertos e (2) os arquivos criados pelo *workflow*. Essas listas são definidas de forma que não sejam aceitos elementos repetidos. Também será declarada uma variável para a identificação do evento de abertura ou criação. A função recebe 2 parâmetros, o primeiro para identificar o programa que será monitorado, ou seja, todos os programas derivados por ele (processos vinculados em uma árvore de execução), e o segundo para informar o diretório do pacote reprodutível (local onde os arquivos monitorados serão armazenados).

O algoritmo faz verificação da ocorrência dos eventos de abertura e criação de arquivos. O algoritmo fica aguardando uma dessas ocorrências. Quando o algoritmo identifica um evento, e verifica o tipo de ocorrência e discrimina o caminho do arquivo manipulado. Quando um evento é do tipo "criação", os dados do arquivo identificado são adicionados à lista de arquivos criados, caso contrário, são armazenados na lista de arquivos abertos. Programas e bibliotecas são arquivos como outro qualquer que são lidos durante a execução de um processo, portanto, são os arquivos envolvidos nesse evento que precisam ser armazenados no pacote reprodutível. Diante disto, o algoritmo cria uma nova lista contendo apenas os arquivos que existiam antes da execução do *workflow*, ou seja, ele desconsidera os arquivos armazenados sob o evento de "criação" e mantém apenas aqueles identificados no evento de "abertura".

O algoritmo cria um novo arquivo para receber todos os arquivos contidos na lista de arquivos final que foram resultado do processo de remoção executado no passo anterior. A partir desse ponto é feita uma leitura nessa lista de arquivos para incluir cada arquivo identificado ao pacote de pesquisa reprodutível. Para cada arquivo informado na lista é verificada a sua existência no sistema de arquivos para que ele possa ser adicionado ao pacote. É importante observar que o *(Strace)* não é a única fonte de dados usada para identificar os arquivos e bibliotecas de um programa. São usados os arquivos que estão listados em *"/proc/PID/maps"* do sistema operacional Linux. O arquivo *maps* lista as regiões de memória e arquivos que um dado programa tem acesso.

4.6.4 Seleção dos Recursos para a Implantação na Nuvem

O compartilhamento de experimentos na nuvem abre a oportunidade para a implantação do experimento em um provedor que oferta recursos a um menor custo financeiro, tanto em termos de armazenamento (publicação) quanto em computação

Algoritmo 4.1 Algoritmo de Identificação de Programas e Bibliotecas

```
1: openedFiles ← []
2: createdFiles ← []
3: event ← NULL
4: função IdentificarProgramas(repromonitor, packagedir)
5:     enquanto (event ← nextEvent(repromonitor))! = NULL) faça
6:         eventType ← getEventType(event)
7:         filePath ← getsFilePathOfEvent(event)
8:         se eventType == "CREATE" então
9:             add(createdFiles, filePath)
10:        senão
11:            add(openedFiles, filePath)
12:        fim se
13:    fim enquanto
14:    resultFilesList ← remove(openedFiles, createdFiles)
15:    tarFile ← initializePackage(packagedir)
16:    para i ← 0 até resultFilesList_length faça
17:        file ← resultFilesList_i
18:        se FileExists(file) então
19:            addInPackage(tarFile, file)
20:        fim se
21:    fim para
        devolve tarFile
22: fim função
```

(reprodutibilidade). É importante destacar que os custos relacionados ao uso da infraestrutura da nuvem é compartilhado entre o autor e o leitor da pesquisa reprodutível. O autor é responsável pelos custos de manter uma imagem de máquina virtual e os conjuntos de dados necessários para a reprodução vinculados aos recursos de armazenamento dos provedores de nuvem. O autor precisa manter uma instância de máquina virtual ativa para permitir o acesso dos leitores e revisores aos objetos de perquisa usados e produzidos durante a execução dos experimentos. O autor é responsável ainda pelos custos de reprodução feitas pelos revisores da pesquisa que serão os responsáveis pela devida avaliação do trabalho, quanto ele for submetido a revistas científicas especializadas. Os interessados na reprodução, neste caso os leitores, são responsáveis pelos custos de reprodução da pesquisa. Esses custo serão contabilizados a partir do momento em que for comandada a reexecução e reimplantação dos recursos para a reprodução.

Diante deste cenário, foi desenvolvido um módulo simples para a classificação das ofertas de recursos da nuvem. O principal objetivo é auxiliar na seleção das melhores ofertas baseado na sua demanda de recursos previamente conhecida. Os monitores de processos e arquivos identificam as características de processamento e armazenamento do experimento. O monitor de processo obtém as informações sobre

a plataforma computacional (CPU, memória principal e a quantidade de núcleos de processamento), enquanto o monitor de arquivos analisa os dados usados e produzidos e, portanto, tem informações para mensurar a quantidade de espaço de armazenamento necessário para a implantação e reprodução da pesquisa reprodutível.

As informações sobre a demanda de recursos ficam registradas nos metadados e dados de proveniência do experimento, e são extraídas para uma estrutura que identifica o perfil dos recursos de computação e armazenamento usados na execução. Essa estrutura pode ser observada na Figura 4.10. As informações sobre os recursos dos computadores originais e instâncias das nuvens são preenchidas utilizando os campos identificados na posição $i = 0$ até aqueles da posição $i = 9$. As demais informações, tais como o nome da instância, provedor de nuvem e preço ($i = 10$ até $i = 12$), são restritas aos recursos obtidos dos repositórios *web* dos provedores de nuvem. Esses outros campos são usados para identificar cada uma das instâncias que serão posteriormente importadas para este módulo.

VCPU	RAM	HD	Number Cores	External Network	Internal Network	Platform	Operational System	Region	Id	Instance	Provider	Price
i=0	*i=1*	*i=2*	*i=3*	*i=4*	*i=5*	*i=6*	*i=7*	*i=8*	*i=9*	*i=10*	*i=11*	*i=12*

Figura 4.10: Estrutura de dados para armazenar as informações dos computadores.

A partir das informações sobre o computador é possível identificar quais as instâncias dos provedores de nuvem possuem recursos mais similares aos usados para a execução do experimento. O principal objetivo é manter a infraestrutura de reprodução o mais próximo da originalmente usada na execução. Portanto, o módulo de análise de ofertas acessa as informações sobre os perfis e valores das instâncias de um grupo de provedores e extrai os dados sobre cada instância para uma estrutura similar a apresentada na Figura 4.10. O resultado dessa extração será uma matriz de ofertas de instâncias contendo os $N = 12$ atributos da estrutura e M instâncias de recursos de computação e armazenamento obtidos dos provedores de nuvem.

Até este ponto, o módulo de análise de ofertas possui o perfil de recursos do sistema de computador $computer_{K,N}$ usado para a execução do experimento, assim como os perfis de todas as instâncias recuperadas dos provedores de nuvem $instances_{M,N}$. A partir deste conjunto de informações, é possível aplicar um método de cálculo de distâncias dos perfis dos recursos $computer_{K,N}$ em relação a $instance_{M,N}$ para classificar quais as ofertas possuem os recursos mais similares com os que foram usados na execução do experimento. Foi implementado um módulo para associar um valor de medida ao conjunto de recursos de cada um dos sistemas de computadores como a Teoria de Utilidade de Multi Atributos (MAUT) [95] e, em seguida, foi aplicada a distância Euclidiana para calcular a similaridade dos recursos ofertados na nuvem em relação aos usados no experimento original.

A Figura 4.11 apresenta um exemplo de distribuição das intancias ofertadas pelos provedores em nuvem (azul) e um computador usado para a execução de um experimento (ponto em vermelho). Neste caso, o recurso computacional em vermelho tem as características de memória RAM equivalente a $26Gbytes$, disco com $16Gbytes$, 4 núcleos de processamento e valor equivalente a US$ $0,113$ a hora. O preço é avaliado em US$. O eixo das abcissas X no gráfico possui os valores financeiros das instâncias (em dólares - US$) e o eixo das ordenadas Y contendo o valor das distâncias. O valor da distância dos recursos, ou seja, as características dos computadores e instâncias foram calculados a partir do algoritmo MAUT.

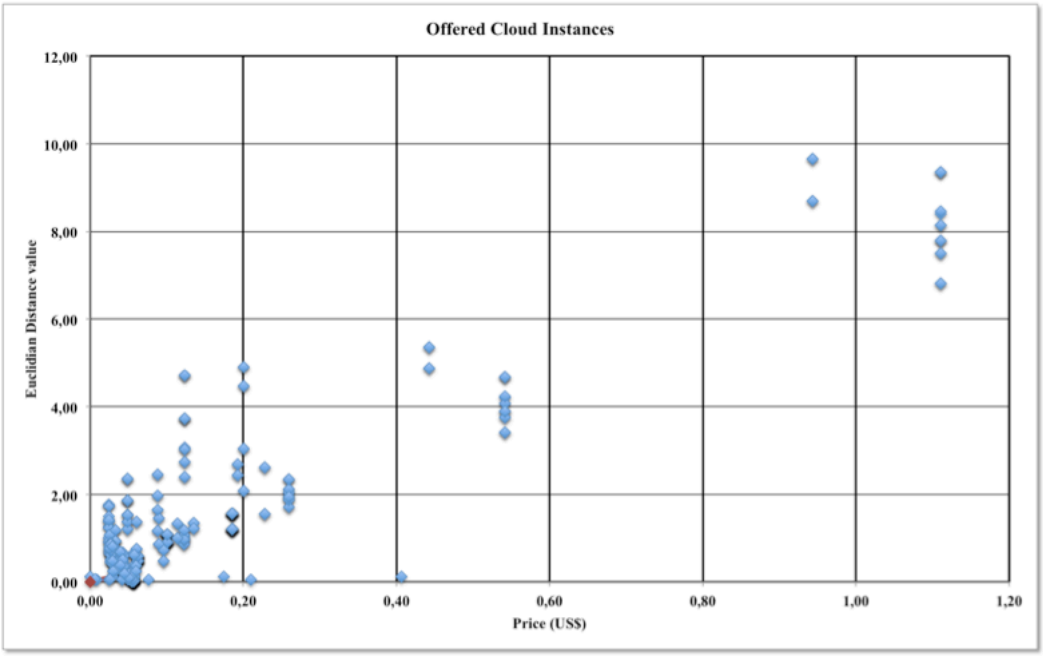


Figura 4.11: Gráfico com o cálculo da distância das características dos recursos (CPU, memória, disco) em relação ao custo US$/hora.

O MAUT é baseado na teoria da utilidade de atributos. Trata-se de uma abordagem sistemática para quantificar preferências de um indivíduo [96]. Ele é usado para fornecer um valor numérico de uma medida de interesse em uma escala de $0-1$ para cada atributo em um conjunto de atributos avaliados, onde 0 representa a pior preferência e 1 a melhor. O algoritmo MAUT (*Multi-Attribute Utility Theory*) ajuda a classificar os recursos oferecidos pelos provedores em relação aos recursos usados para executar o *workflow* original. Ele associa uma determinada medida a computador original e a cada uma das instâncias dos provedores de nuvem. A partir desse ponto é possível aplicar um algoritmo para calcular a distância dos recursos da nuvem em relação à infraestrutura original.

A Equação 4.1 apresenta a função de utilidade $U(x_1, \ldots, x_N)$, onde o valor U_i representa os valores de utilidade de algum atributo $i = 0, \ldots N$. Os valores x_i representam cada uma das características computacionais avaliadas (frequência de

ciclos de processador, *bytes* de armazenamento, bits/segundos de transferência etc). Considerando as análises executadas para este artigo, esses atributos representariam os valores velocidade de CPU, quantidade de memória, vazão da rede etc. A variável w_i representa o valor do peso da preferência desse atributo. Este valor é determinado pelo cientista para priorizar sua preferência de uma característica em relação a outra (i.e. prefere ter maior banda de rede do que processamento). Ao final da execução, a função MAUT apresenta uma classificação ordenada de alternativas que reflete as preferências dos decisores.

$$U_{j=1,\dots M} = \sum_{i=1}^{4} (w_i * U_i * (x_i)) \tag{4.1}$$

Os valores de medida foram aplicados somente aos $N = 4$ primeiros atributos dos vetores das ofertas. O valor da utilidade pode ser obtido por meio da normalização dos atributos avaliados. A equação 4.2 apresenta o cálculo aplicado para normalizar os valores, obtendo-se o maior e o menor valor do domínio de valores do atributo avaliado (instâncias da nuvem e computador usado no experimento). Esse procedimento permite analisar os atributos nas de escalas corretas valores. Esse procedimento inclui o cálculo da medida de utilidade U também para os computadores usados no experimento original.

$$Z_i^k = \frac{Z_i^k - Z_{min}^k}{Z_{max}^k - Z_{min}^k} \tag{4.2}$$

Após um valor de medida de utilidade ter sido calculado, para todas os computadores do ambiente de produção e das instâncias de recursos da nuvem, são extraídos os valores das ofertas das M instâncias. Os computador da infraestrutura de origem possui o valor equivalente a 0, ou seja, $X_1 = 0$. Os valores relacionado aos preços de cada instância serão relacionados a X_2, onde $X_2 = instance_{j,12}, j = \{j = 1 \dots M\}$, ou seja, a posição $i = 12$ (vetor apresentado na Figura 4.10) possui a informação do preço da instância. O valor da medida de utilidade do computador original será expesso como $Y_1 = U_p, p = \{1, \dots, K\}$ e o valor de utilidade de uma das instâncias $Y_2 = U_j, j = \{j = 1 \dots M\}$. A partir desse conjunto de valores é aplicado o cálculo da distância Euclidiana para cada instância de oferta da nuvem em relação aos recursos do experimento original. A Equação 4.3 apresenta o cálculo da distância.

$$distance = \sqrt{(X_1 - X_2)^2 + (Y_1 - Y_2)^2} \tag{4.3}$$

A Figura 4.12 apresenta alguns exemplos de ofertas extraídas dos provedores de nuvem usados no desenvolvimento deste trabalho. Tais ofertas já estão organizadas conforme uma classificação aplicada com os métodos usados no exemplo proposto nesta seção.

Provider	Id	Graph Id	Price	Memory RAM	Hard Disk	Number of Cores	Virtual CPU	External Network	Internal Network	Platform	Operation System	Region	Control
Azure	A0	a200	$0.018	0.75	20.0	1.0	1.0	1.0	1.0	64	Windows	East US	Select
Azure	A0	a230	$0.018	0.75	20.0	1.0	1.0	1.0	1.0	64	Windows	East US	Select
Azure	A0	a260	$0.018	0.75	20.0	1.0	1.0	1.0	1.0	64	Linux	East US	Select
Azure	A0	a290	$0.018	0.75	20.0	1.0	1.0	1.0	1.0	64	Linux	East US	Select
Azure	A0	a205	$0.02	0.75	20.0	1.0	1.0	1.0	1.0	64	Windows	East US	Select
Azure	A0	a235	$0.02	0.75	20.0	1.0	1.0	1.0	1.0	64	Windows	East US	Select
Azure	A0	a265	$0.02	0.75	20.0	1.0	1.0	1.0	1.0	64	Linux	East US	Select
Azure	A0	a295	$0.02	0.75	20.0	1.0	1.0	1.0	1.0	64	Linux	East US	Select
Azure	A1	a231	$0.044	1.75	40.0	1.0	1.0	1.0	1.0	64	Windows	East US	Select
Azure	A1	a291	$0.044	1.75	40.0	1.0	1.0	1.0	1.0	64	Linux	East US	Select

1 2 3 4 5 6 7 8 9 10

Figura 4.12: Exemplo de descrição de ofertas de instâncias na nuvem.

O ReproeScience aplica a Distância Euclidiana e o MAUT para auxiliar o cientista na análise e classificação de ofertas de recursos da nuvem. Primeiramente faz essa classificação para identificar o menor custo para armazenar uma VM do computador usado na execução do *workflow* na área de armazenamento de um provedor de nuvem. Este armazenamento é feito para fins de publicação e compartilhamento do experimento. Em seguida, classifica as ofertas de recursos de computação para a reprodução do experimento. Em geral, envolve a análise das características operacionais dos computadores, tais como a unidade de processamento, memória principal, armazenamento em massa, arquitetura do hardware, sistema operacional e vazão de rede, de forma a selecionar o melhor custo/benefício para implantar e infraestrutura do experimento e reproduzir o *workflow*.

4.6.5 Implantação da Pesquisa Reprodutível

Ao final da execução do *workflow*, todos os procedimentos para coleta de informações são finalizados e os dados gerados com a execução do experimento são devidamente armazenados na base de proveniência do ReproeScience. A partir desse momento inicia-se a implantação do ambiente de reprodução. Trata-se da etapa responsável por preparar a VM que vai hospedar a pesquisa reprodutível. É nessa etapa que a infraestrutura dos computadores serão empacotados, enviados e implantados em um provedor de nuvem. O ReproeScience cria uma VMI e, em seguida, implanta os objetos de pesquisa e base de dados com as informações sobre o experimento compartilhado para a reprodução. Existem duas formas de gerar essa VMI: (1) na estação de trabalho do cientista por meio dos MMVs, ou (2) na estrutura da nuvem, usando as ferramentas disponibilizadas pelos próprios provedores de nuvem.

No primeiro caso, é necessário utilizar um *hypervisor* como o *VMWare Virtualization for Desktop and Server* (http://www.vmware.com) ou *Oracle Virtual Box* (https://www.virtualbox.org/) para transformar a estação de trabalho ou os nós do *cluster* dos cientistas em VMIs. É necessário usar três sistemas de computadores, dois deles podem ser VMs sob o computador físico do experimento. O primeiro computador acomoda a infraestrutura do experimento, o segundo é responsável por gerenciar um conversor do computador físico para uma VM e o terceiro receberá a VM convertida para VMI. O computador conversor utiliza ferramentas de conversão dos *hypervisors* para transformar máquinas físicas em VMs. Ele estabelece uma conexão no computador do cientista através do protocolo SSH (*Security Shell*) e cria uma VM auxiliar no computador de destino. Em seguida, identifica os elementos a serem recuperados e obtém o sistema operacional, programas e os dados do computador de origem e os implanta nessa VM auxiliar. Ao final, o conversor conecta no computador de destino e configura a VM auxiliar para permitir o processo de

inicialização da VM pelo sistema operacional do computador de destino.

Essa primeira forma de VMI acomoda bem os casos em que o cientista deseja compartilhar a infraestrutura para ser executada diretamente em outra estação de trabalho ou *cluster*, porém, pode apresentar alguns problemas na implantação no provedor de nuvem. O motivo é que essas VMIs não possuem os agentes e aplicativos necessários para o controle das rotinas de serviço de fornecimento de recursos definidos pelo provedor de nuvem. Portanto, quando uma VMI é criada na estação de trabalho local e, posteriormente, é enviada para nuvem, é necessário que o provedor faça as suas adaptações, e esse processo pode acarretar erros impossibilitando a utilização dessa VMI na nuvem. Boa parte dos testes realizados na concepção da abordagem foram bem sucedidos, porém, uma parcela mínima (em torno de 5%) apresentou esse tipo de problema.

Já o segundo caso, a criação das VMIs é feita diretamente no provedor de nuvem, através de Applications Programming Interface (APIs) ou protocolos como o REpresentational State Transfer (REST) fornecidos pelos provedores. Neste segundo caso, o cientista seleciona o provedor de nuvem que deseja implantar a VMI do experimento e um tipo de instância dentre as oferecidas pelo provedor, geralmente guiado por aquelas que possuem as características computacionais mais similares com a infraestrutura que o cientista utilizou para executar o seu *workflow* utilizando o cálculo de distância dos recursos usados na execução original em relação as instâncias ofertadas pelos provedores de nuvem.

É importante observar que os recursos ofertados pelos provedores de nuvem possuem características operacionais pré-estabelecidas a um determinado custo, ou seja, quaisquer que sejam as características operacionais dos sistemas utilizados na execução do experimento, ao ser enviado para a nuvem, devem atender as características das instâncias ofertadas pelo provedor. Por esse motivo não é possível obter recursos de computação exatamente iguais aos utilizados no experimento original. Com isso, é necessário que o cientista selecione as ofertas de recursos mais parecidas possível com aquelas que ele utilizou em sua estação de trabalho.

O Algoritmo 4.2 apresenta todas as etapas seguidas pelo ReproeScience para a implantação, publicação e compartilhamento de uma pesquisa na nuvem. Foi focada a segunda estratégia de implantação pelo fato de necessitar de adaptações na reconstrução do ambiente. O algoritmo recebe 4 parâmetros: (1) *provider*, com os dados de descrição e endereço *web* do provedor de nuvem, (2) credentials, contendo as informações das credenciais do interessado em compartilhar a pesquisa reprodutível, (3) *wfdirectory*, que representa o diretório do *workflow* na máquina de destino, e (4) *dbconnection*, sendo os dados sobre a conexão com a base de dados que será implantada na VM.

O algoritmo faz a compressão do diretório que o ReproeScience armazenou os

objetos de pesquisa do experimento. Nesse momento é criado um pacote contendo todos os dados comprimidos para a redução do tamanho antes da transferência. É feito um *backup* dos dados, metadados e proveniência do banco de dados do ReproeScience para que ela seja recuperada e reimplantada posteriormente na nuvem. Nesse momento são informados o nome da base de dados, serviço, porta e a localização. Em seguida, o algoritmo obtém os tamanhos do diretório do *workflow* no computador de origem e do pacote comprimido com os objetos de pesquisa, e ainda, mais $100Mbytes$ para ter um espaço extra. Esse processo é executado para mensurar o espaço necessário para armazenar os dados para a implantação do experimento na nuvem. Até esse ponto, o algoritmo concentrou os esforços para identificar os pré requisitos para preparar o envio dos objetos de pesquisa para a nuvem, entretanto, a partir desse ponto inicia-se a transferência e implantação da pesquisa na nuvem.

A primeira ação do algoritmo na nuvem é fazer o registro junto ao provedor para obter acesso ao leque de recursos oferecidos. Para isso, é necessário passar as informações do provedor e das credenciais. Após registro, o algoritmo aloca o recurso de VM que demanda o menor custo, definida aqui como "*Small Machine*". Essa definição foi adotada pelo fato dessa VM ser responsável apenas pela publicação e compartilhamento dos dados e não para a tarefa de reprodução. Serão escolhidos os recursos mais adequados para a reprodução baseado nos dados de proveniência obtidos durante o monitoramento da execução. Após implantada a VM é feita uma conexão remota para enviar o pacote do experimento, arquivo de instalação dos *softwares*, base de dados, e instalação e implantação da pesquisa reprodutível.

Se o espaço total de armazenamento do experimento for menor ou igual a $3Gbytes$ (atual capacidade das instânicas de máquinas virtuais a nuvem) é possível implantar os dados nos discos da própria VM instanciada para receber os pacotes da pesquisa reprodutível nos provedores testados neste trabalho. Nesse caso, é feita a criação e definição do diretório do *workflow* na máquina de destino via conexão remota. Entretanto, se o espaço requerido for superior a $3Gbytes$ é necessário alocar espaço no sistema de armazenamento da nuvem, ou seja, será preciso adicionar um disco extra aos recursos alocados. Para enviar a infraestrutura para a nuvem é preciso autenticar com os dados de acesso do cientista e, em seguida, comandar os processos de gerenciamento dos recursos disponibilizados pelo provedor. Após o registro, o ReproeScience cria os recursos de computador e armazenamento necessários para a publicação e compartilhamento da pesquisa. Ele implanta o sistema operacional ofertado na nuvem e, ao final, transfere todos os elementos usados na execução do *workflow* para essa VM via protocolo SSH (*Secure Shell*).

Quando o processo de envio da infraestrutura é finalizado, deve ser definido o tipo de compartilhamento desses recursos para os demais usuários da nuvem. Deve ser informado ao provedor se as VMs do experimento, a partir desse momento deno-

Algoritmo 4.2 Algoritmo de Implantação

1: $package$ = Pacote com objetos de pesquisa do *workflow* comprimidos.
2: $dbbackup$ = Arquivo com as instruções para recuperação dos dados da pesquisa.
3: $wfdirectorysize$ = Registra o espaço necessário para armazenar o *workflow*.
4: $wfpackagesize$ = Registra a quantidade de espaço necessário para o armazenamento do pacote do experimento.
5: $requiredspace$ = Armazena espaço necessário para a implantação.
6: $cloudservice$ = Registra os dados do serviço alocado na nuvem.
7: $vmmetadata$ = Máquina virtual que receberá a pesquisa reprodutível.
8: $remoteconnection$ = Armazena os dados da conexão com a VM na nuvem.
9: **função** IMPLANTAÇÃO(provider, credentials, wfdirectory, dbconnection)
10: $package \leftarrow comprimir(wfdirectory)$
11: $dbbackup \leftarrow executeDBBackup(dbconnection)$
12: $wfdirectorysize \leftarrow getFileSystemSize(wfdirectory)$
13: $wfpackagesize \leftarrow getFileSystemSize(package)$
14: $requiredspace \leftarrow wfdirectorysize + wfpackagesize + 100_{Mbytes}$
15: $cloudservice \leftarrow cloudServiceBuild(provider, credentials)$
16: $vmmetadata \leftarrow createVirtualMachine(cloudservice, "SmallMachine")$
17: $remoteconnection = getRemoteConnetion(vmmetadata)$
18: **se** $requiredspace <= 3_{Gbytes}$ **então**
19: $mkdir(remoteconnection, wfdirectory)$
20: **senão**
21: $requiredspace = IntegerOf(requiredspace) + 1_{Gbyte}$
22: $createExtraDisk(cloudservice, wfdirectory, requiredspace)$
23: **fim se**
24: $installBasicSoftware(remoteconnection)$
25: $uploadFile(remoteconnection, wfdirectory, wfpackagesize)$
26: $uploadFile(rremoteconnection, "/tmp", dbbackup)$
27: $extractFile(remoteconnection, wfdirectory, package)$
28: $restoreDataBase(remoteconnection, "/tmp", dbbackup)$
29: **fim função**

minadas VMIs, devem ser compartilhadas publicamente, ou se devem ser utilizadas apenas por alguns usuários específicos. Em geral, o sistema operacional dessa nova VM mantém os usuários que foram especificados na infraestrutura do experimento original, todavia, reinicializa a senha de acesso, sendo necessária a definição de uma nova senha na criação de cada nova instância.

O ReproeScience gera um valor de *hash* para cada objeto de pesquisa associado com a execução e armazena esses dados na base de proveniência do *workflow* científico. Esse processo tem como objetivo garantir a integridade do conteúdo que está sendo compartilhado. A implantação de fato será iniciada somente após a verificação da equivalência dos valores dos *hashs* dos objetos de pesquisa do experimento com os objetos de pesquisa copiados para a nuvem. É uma etapa de conferência, que tem o objetivo de verificar se todos os elementos do experimento foram corretamente enviados para a nuvem. Portanto, quando o pacote reprodutível for totalmente copiado, todos os elementos são extraídos na VM de destino em um mesmo diretório de origem, a exemplo do computador do experimento original.

O processo de restauração do pacote cria um diretório do ReproeScience nas VMs que vão reproduzir a pesquisa. Esse diretório é considerado o novo diretório raiz sistema operacional. Todos os arquivos identificados no monitoramento são extraídos para esse diretório. Essa definição permite a reexecução de todos os elementos usados na produção dos resultados (dados, programas, variáveis de ambiente). O pacote reprodutível cria um exemplo de ambiente isolado somente com os elementos do experimento obtidos pelos monitores do ReproeScience.

O Algoritmo 4.6.5 apresenta as etapas necessárias para a retauração do pacote reprodutível. Ele recebe 3 parâmetros: (1) o diretório do *workflow* científico, (2) o caminho para o diretório do pacote reprodutível e (3) e um sinalizador com a opção de montar ou não um disco externo. A definição do disco externo ocorre nos casos que o disco fica em um local remoto ou nas situações em que o disco local não tem espaço suficiente para armazenar os dados produzidos pelo experimento. O algoritmo inicia a execução extraindo os elementos do pacote para o diretório raiz da pesquisa reprodutível denominado como "/ReproExperiment'/". Esse diretório agora será o novo diretório raiz do sistema operacional. Se não for preciso montar um disco externo o algoritmo finaliza a execução.

Algoritmo 4.3 Algoritmo de restauração do pacote reprodutível.

```
1: função RestauraPacote(wfdirectory, packagefilepath, mountexternaldisk)
2:     extractFile(PackageFilePath, "/ReproExperiment/")
3:     se MountExternalDisk então
4:         correctwfdir ← "/ReproExperiment/" + WorkflowDir
5:         mkdir("/tmp/workflowDir")
6:         mv(correctwfdir, "/tmp/wfdirectory")
7:         reproMountDisk(correctwfdir)
8:         mv("/tmp/wfdirectory", correctwfdir)
9:     fim se
10: fim função
```

Se for necessário montar um disco externo é preciso identificar o caminho correto do *workflow*, após extraído a partir do novo diretório raiz. Logo após, é feita a criação de um diretório temporário para acomodar os arquivos do *workflow* temporariamente enquanto o disco que irá receber os arquivos do *workflow* é montado. É importante observar que essa operação ocorre apenas na implantação do nó principal do ambiente paralelo. Os demais nós apenas montar uma pasta compartilhada que representam esse diretório do disco. Ao final, após o disco ser montado, todos os arquivos são movidos do diretório temporário para o diretório do *workflow*.

A portabilidade é garantida por meio do monitoramento dos processos invocados em tempo de execução e dos dados de proveniência armazenados na base de dados. São identificados os programas e bibliotecas invocados na execução do *workflow* durante a fase de monitoramento. Esses programas são listados para o cientista para auxiliá-lo na preparação das diretrizes para a reinstalação dos programas necessários para a execução do experimento. A Figura 4.13 apresenta o arquivo JSON contendo os comandos necessários para a instalação dos programas do ReproeScience. Esses comandos instalam os aplicativos necessários para a execução da abordagem.

A primeira definição é a instalação do pacote java (notações *software* e *java*), uma das linguagens na qual o ReproeScience foi desenvolvido. Logo após, é definida a implantação do servidor de banco de dados (anotação *server*) que irá armazenar a base de dados, metadados e proveniência. Em seguida, são definidos os aplicativos utilitários, apresentados na anotação *commons*. Como a abordagem apoia a reprodução de ambientes distribuídos, é necessário adotar uma forma de implantar o ReproeScience em todos os nós de processamento em ambiente de HPC. A princípio, poderiam ser adotadas duas estratégias, a primeira, manter os programas em um diretório compartilhado, ou no segundo caso, fazer uma cópia dos programas para todos os computadores usados na execução. Por questões de desempenho foi adotada a segunda estratégia, expressa por meio da anotação *export-software*. A última

```
lgor{
  "software": {
    "java": {
      "RunCMD": "echo oracle-java8-installer shared/accepted-oracle-license-v1-1 select
true | sudo /usr/bin/debconf-set-selections",
      "apt-add-repository": "ppa:webupd8team/java",
      "apt-update": true,
      "apt-install": "oracle-java8-installer"
    },
    "server":{
            "posgres": {
              "apt-install": [
                "postgresql",
                "postgresql-contrib"
              ]
            }
    },
    "Commons": {
      "apt-install": [
        "zip",
        "unzip",
        "gcc",
        "make",
        "htop"
      ]
    },"MPJ":{
        "export-var":"MPJ_HOME=$REPRO/mpj",
        "export-bin":"$MPJ_HOME/bin"
    }
  },
  "export-software": {
    "ReproSoft": {
      "in-shared-folder":false,
      "first":{
        "RunCMD": "sh compileStrace.sh"
      },
      "DIR":".",
      "export-var": "REPRO=$PWD",
      "export-bin": "$PWD/bin",
      "RunCMD":"cd $PWD/bin/;chmod 777 *"
    }
  },
  "database": {
    "createDB": "repro",
    "change-password": {
      "user-name": "postgres",
      "password": "1234"
    },
    "enable-remote-access": true,
    "restore-sql": {
      "database-name": "repro",
      "file": "repro.sql"
    }
  }
}
```

Figura 4.13: Arquivo JSON com as diretrizes de instalação.

definição é em relação ao banco de dados (anotação *database*). Nessa definição, o banco de dados, usuário e demais objetos são criados no servidor e, a partir desse ponto, o ambiente do ReproeScience está configurado e apto para o uso.

Para portar o experimento, o componente analisa as questões de compatibilidade dos ambientes operacionais em diferentes camadas, iniciando da infraestrutura de *hardware*, seguida pelo sistema operacional, passando pelos programas e biblioteca, para que, ao final, sejam verificados os dados e demais arquivos. Os metadados e proveniência orientam a implantação desses elementos. Primeiramente, o ReproeScience obtém as características operacionais para a criação dos recursos de computação e armazenamento a partir da proveniência, portanto, a quantidade de computadores, infraestrutura de redes e o armazenamento. Em seguida, na camada de sistemas

operacionais, orienta na seleção do sistema com a base operacional idêntica e com versões equivalentes ou superiores para a implantação.

4.6.6 Reprodução do Experimento

Para reproduzir o experimento é necessário fazer a assinatura de um serviço de nuvem. O ReproeScience foi testado em 3 provedores, Amazon Web Services[2], Google Cloud Plataform[3] e Microsoft Azure[4], e está apto a reimplantar e reproduzir as pesquisas reprodutíveis em qualquer uma dessas infraestruturas. Em alguns casos, é necessário obter uma chave de acesso que deve ser implantada no computador do usuário. Mesmo que uma pesquisa científica tenha sido definida em um provedor de nuvem é possível migrar para outros que possuem o suporte do ReproeScience.

A etapa de reprodução permite que o *workflow* compartilhado na nuvem possa utilizar um conjunto de recursos especializados da nuvem para permitir testar diferentes parâmetros e conjuntos de dados. A partir dessa etapa, é possível alterar a estrutura de execução do *workflow*, criando uma nova versão, ou ainda reutilizando alguns pedaços do processo em novos *workflows*. O ReproeScience poderá dar o suporte para a reprodução independente de alterações realizadas no *workflow*.

O Algoritmo 4.6.6 apresenta as etapas necessárias para a reimplantação e execução do experimento na nuvem via ReproeScience. O algoritmo recebe 7 parâmetros: (1) *provider*, que possui os dados do provedor de nuvem, (2) *credentials*, com as credenciais para uso dos recursos dos provedores de nuvem, (3) *wfdirectory*, sendo o diretório de trabalho do *workflow*, (4) *vmtemplatelist*, representando cada um dos *templates* de VMs que serão implantadas para reproduzir o experimento, ou seja, a estrutura com o modelo para instanciação das VMs, (5) *package*, tratando-se do pacote contendo os objetos de pesquisa usados na execução original, (6) *requiredspace* é o espaço necessário para reimplantar o experimento, e (7) *vminstancetype*, que define os tipos de instâncias que serão usadas para a alocação das VMs.

O algoritmo inicia com a autenticação e confirmação dos recursos no provedor de nuvem para iniciar a construção e implantação dos serviços de reprodução. Em seguida, o nome do pacote do experimento é obtido para que posteriormente o diretório do experimento seja implantado nas VMs. A reimplantação trata os ambientes compostos de 1 nó de processamento de forma diferente daqueles compostos por vários. O componente faz uma verificação no número de *templates* de VMs armazenadas na base de proveniência. Caso o número de *templates* seja menor que o valor 1, trata-se de uma arquitetura com apenas um nó de processamento, caso contrário, vão existir diversos computadores, até mesmo com características heterogênea. Cada um desses

[2]www.aws.amazon.com
[3]https://cloud.google.com/compute/
[4]http://azure.microsoft.com/pt-br/

casos precisa de um tratamento diferente na alocação das instâncias de VMs.

Quando a quantidade de *templates* é maior que 1 é preciso definir qual dos nós implantados será o nó servidor. Esta definição é feita para se obter as informações de referência do repositório obtendo o endereço IP e as devidas portas de serviços. Após a criação do nó servidor é preciso fazer uma conexão, primeiramente, para a criação dos discos necessários para armazenar os dados do experimento e, em seguida, para carregar e extrair dos objetos de pesquisa contidos no pacote do experimento para o diretório do *workflow* na VM que representa o nó servidor. Ao final, é preciso compartilhar esse diretório com todos os nós participantes, portanto, é feita a instalação e ativação do servidor de arquivos compartilhados.

Após terminar a implantação do nó servidor inicia-se a construção dos nós clientes. Cada VM cliente é instanciada conforme as estruturas informadas no *template*. Essas VMs são associadas aos serviços de nuvem que estão vinculados ao interessado na reprodução. O ReproeScience obtém a conexão com cada uma dessas VMs e armazena essas conexões na sua base de dados. Como o ReproeScience apoia a reprodução de experimentos de ambientes HPC, é preciso implantar um sistema de arquivos paralelo e distribuído. Essa implantação deve ocorrer nos mesmos moldes que a efetuada no servidor, fazendo o compartilhamento do sistema de arquivos dos clientes e informando o endereço IP do servidor. Ao final, são instalados todos os *softwares* informados no arquivo JSON apresentado na Figura 4.13.

No caso dos experimentos realizados com apenas um sistema de computador, é feita a criação de uma VM usando a única descrição de computador contida na lista de *templates* sob os serviços de nuvem vinculados a credencial. Em seguida, é feita uma conexão para o nó recém criado para fazer a criação do disco que irá receber o pacote com os objetos de pesquisa. Ainda com a conexão, são executadas a implantação e extração do pacote de pesquisa reprodutível. Após esse ponto, o ReproeScience executa a configuração dos seus mecanismos, instala e configura a VM com base nas informações contidas no arquivo JSON.

Para reproduzir um experimento com características de paralelismo e distribuição de recursos é necessário fazer a implantação de um sistema de arquivo com acesso distribuído. Isso se deve ao fato do ReproeScience apoiar a reprodução de *workflows* distribuídos, que atuam em vários nós para o processamento das atividades do experimento. Ao processar uma tarefa, a atividade não pode ter acesso somente ao seu próprio contexto/ambiente, porém, deve ter acesso a todos os arquivos e programas dos outros nós. Essa demanda acarreta na necessidade de usar um sistema de arquivo distribuído com uma boa taxa de transferência e baixa latência, de modo que a velocidade de acesso aos arquivos não seja um gargalo para a execução do *workflow*. É necessário usar um sistema de arquivo específico somente no caso do *workflow* trabalhar diretamente com a manipulação de arquivos, em casos em que

Algoritmo 4.4 Algoritmo de Reimplantação Para a Reexecução

1: $cloudservice$ = Registra os dados do serviço alocado na nuvem.
2: aux = variável auxiliar que receberá o nome do pacote do *workflow*.
3: **função** REIMPLANTAR(provider, credentials, wfdirectory, vmtemplatelist[], package, requiredspace, vminstancetype)
4: $cloudservice \leftarrow cloudServiceBuild(provider, credentials)$
5: $clusterinfo \leftarrow NULL$
6: **se** $vmtemplatelist\,[]\,.size > 1$ **então**
7: $servernode \leftarrow createVirtualMachine(cloudservice, vminstancetype)$
8: $serverip \leftarrow getIpOfMachine(servernode)$
9: $connection = getConnetion(servernode)$
10: $createExtraDisk(cloudService, servernode, wfdirectory, requiredspace)$
11: $enableSharedFolderServer(connection, wfdirectory)$
12: $uploadFile(connection, wfdirectory, package)$
13: $extractFile(connection, wfdirectory, package.name)$
14: $nodes[] \leftarrow createVM(cloudservice, vmtemplatelist\,[])$
15: $connectionNodesList[] \leftarrow getConnetion(nodes[])$
16: $startSharedFolderClient(connectionNodesList[], serverip)$
17: $setupReproSoftware(servernode, nodes)$
18: $setupMachinesFromJsonFile(servernode, nodes, wfdirectory, \text{“}conf.json\text{”})$
19: $clusterInfo \leftarrow mountClusterInfo(servernode, nodes)$
20: **senão**
21: $node \leftarrow createVM(cloudservice, vmtemplatelist\,[0])$
22: $connection = getRemoteConnetion(node)$
23: $createExtraDisk(cloudService, servernode, wfdirectory, requiredspace)$
24: $uploadFile(connection, wfdirectory, package)$
25: $extractFile(connection, wfdirectory, package.name)$
26: $setupReproSoftware(node)$
27: $setupMachinesFromJsonFile(node, wfdirectory, \text{“}conf.json\text{”})$
28: $clusterInfo \leftarrow mountClusterInfo(node)$
29: **fim sedevolve** $ClusterInfo$
30: **fim função**

esse arquivo poderá ser usado por qualquer um dos outros nós processadores.

Foi adotado o sistema de arquivos paralelo e distribuído denominado BeeGFS[5] por ter um foco nas características de desempenho e conveniência na instalação e gerenciamento. Além disso, prevê cargas de trabalho intensivas para operações de entrada e saída de dados (I/O). O sistema de arquivos permite escalar os recursos de computação e armazenamento para se obter um melhor desempenho e maior capacidade. A adoção dos sistemas de arquivos paralelos e distribuídos foi fundamental para garantir o desempenho na reprodução dos *workflows* científicos, conforme será discutido no próximo Capítulo.

4.6.7 Avaliação: Isomorfismo dos Grafos de Proveniência

A análise da reprodutibilidade dos *workflows* é aplicada sob os metadados e dados de proveniência gerados a partir das execuções dos *workflows* científicos. Cada execução do *workflow* gera novos registros de metadados e proveniência formando os dados relativos à execução desse ensaio. Estes registros são inseridos no final do arquivo W3C Prov XML e são estruturados conforme as definições do tipos e relacionamentos (esses elementos são mostrados no Apêndice A).

A análise da reprodutibilidade é aplicada em pares de ensaios de um *workflow*. Ela verifica se a execução de um ensaio no tempo t produziu os mesmos metadados e dados de proveniência de um ensaio realizado no tempo t'. Essa análise é executada neste trabalho através da aplicação da teoria dos grafos e das suas propriedades de equivalência e isomorfismo. Cada ensaio é representado como um grafo G, denominado aqui como grafo de proveniência. O grafo de proveniência G é construído a partir da extração de todos os tipos e relacionamentos gerados a partir da execução de um ensaio. Ele é composto por um conjunto finito de vértices V e arestas E.

Os tipos de dados e relacionamentos W3C Prov gerados com a execução de um ensaio são representados, respectivamente, pelos vértices V e pelas arestas E do grafo G. São definidos três tipos de vértices V: (1) vértice agente, (2) vértice atividade e (3) vértice entidade. São criados ainda sete tipos de arestas E, baseadas nos relacionamentos definidos pelo W3C Prov: (1) *WasGeneratedBy*, (2) *Used*, (3) *WasInformedBy*, (4) *WasDerivedFrom*, (5) *WasAttributedTo*, (6) *WasAssociatedWith* e (7) *ActedOnBehalfOf*. As arestas *WasInformedBy*, *WasDerivedFrom* e *ActedOnBehalfOf* ligam um vértice a ele mesmo $\{u, u\}$ criando uma ligação denominada *loop*. As demais arestas ligam dois vértices diferentes $\{u, v\}$.

A análise da equivalência é aplicada nos vértices que possuem o mesmo tipo de dado W3C Prov, ou seja, um vértice agente V_{Hag} é comparado com outro vértice agente V_{Gag}, uma atividade V_{Ha} com atividade V_{Ga} e entidade V_{Hag} com entidade

[5]http://www.beegfs.com/content/

V_{Ge}. De forma análoga, as arestas também consideram os mesmos parâmetros, sendo comparados apenas os mesmos relacionamentos dos grafos de proveniência. A função de análise da reprodutibilidade aplica o mapeamento entre vértices e arestas baseadas nessas restrições de tipos e relacionamentos para calcular o isomorfismo dos grafos de proveniência.

O isomorfismo entre dois grafos G e H é denotado por $G \cong H$. Os grafos isomorfos possuem o mesmo número de vértices e cada vértice v possui exatamente o mesmo número de vizinhos. A propriedade do isomorfismo é garantida por uma bijeção dada por uma função $f : V_G \to V_H$ que faz o mapeamento de um grafo para o outro. Considerando que G e H são dois grafos simples, uma função de vértice $f : V_G \to V_H$ preserva a adjacência se para cada par de vértice adjacente u e v no grafo G, os vértices $f(u)$ e $f(v)$ são adjacentes no grafo H. Os grafos isomorfos podem ter os nomes dos vértices v e arestas e diferentes, porém, devem manter uma equivalência estrutural.

Cada vértice v deve ter exatamente o mesmo número de vizinhos em ambos os grafos $G\{V, E\}$ e $H\{V, E\}$. Eles têm o mesmo conjunto de vértices, os quais são mapeados através da função $f : V_G \to V_H$ que preserva a adjacência se todos os pares de vértices adjacentes u e v no grafo G possui os vértices $f(u)$ e $f(v)$ adjacentes no grafo H. De forma similar, f preserva a **não-adjacência** se $f(u)$ e $f(v)$ não forem adjacentes, assim como u e v também não são adjacentes. Uma função bijetora $f : V_G \to V_H$ entre dois grafos G e H preserva a estrutura se: (1) o número de arestas (mesmo em grafos sumidouros) entre todos os pares de vértices distintos u e v no grafo G é igual ao número das suas imagens $f(u)$ e $f(v)$ no grafo H e (2) o número de *loops* de cada vértice x no grafo G for igual ao número de *loops* no vértice $f(x)$ em H.

Dois grafos são **isomorfos** se existe uma função de bijeção de vértice $f : V_G \to V_H$ que preserva a estrutura do grafo. Os pares de grafos isomorfos devem respeitar um conjunto de propriedades que são mostradas nos itens a seguir:

- Sejam G e H grafos isomorfos. Então, eles devem ter o mesmo número de vértices e arestas. O isomorfismo deve mapear V_G e E_G bijetivamente.

- Seja $f : V_G \to V_H$ um grafo isomorfo e seja vV_G. Então, o grau de $f(v)$ deve ser igual ao grau de v. Se f está preservando uma estrutura, o número de arestas e o número de *loops* incidentes sob o vértice v corresponde aos números para o vértice $f(v)$. Desta forma, o grau de $f(v)$ é igual ao grau de v.

O *workflow* é composto por um conjunto de ensaios, e cada ensaio produz um grafo contendo um conjunto de metadados e dados de proveniência obtidos durante a execução do ensaio. A avaliação da reprodutibilidade é feita comparando

o grafo de proveniência da última execução, em relação ao grafo da execução do primeiro ensaio. Portanto, os grafos G e H representam, respectivamente, os grafos de proveniência do experimento original G e do reproduzido H. Novas execuções do *workflow* geram novos grafos de proveniência H que são comparados com G. Esta operação permite identificar se os dois ensaios fizeram o mesmo fluxo de execução de atividades e também quais foram os conjuntos de dados consumidos e produzidos pelas atividades.

Se o grafo de proveniência de ambos os grafos G e H forem isomorfos $G \cong H$, significa que a reprodução foi totalmente executada. Ele atesta que todas as atividades foram completamente executadas segundo sua ordem de precedência, confirma a utilização e produção de todos os arquivos, parâmetros e sistemas de computadores por cada uma das atividades, enfatiza todas as derivações de entidades identificadas durante a produção de novas entidades. Portanto, significa que a reprodução dos resultados e trilha da execução foi 100% alcançada.

Muitos *workflows* científicos produzem arquivos intermediários baseado no conteúdo de arquivos executados em atividades antecessoras e nos parâmetros de configuração informados para a execução da atividade. Portanto, mesmo usando o mesmo conjunto de dados, a produção de arquivos intermediários pode ser diferente em duas execuções distintas de um mesmo experimento, em períodos diferentes, t e t'. Isto se deve ao fato de que grande parte dos experimentos computacionais são aplicados a domínios aleatórios ou em domínios que manipulam valores em ponto flutuante e, portanto, podem gerar resultados numéricos diferentes, muitas vezes devido a heurística adotada para a produção dos algoritmos ou, até mesmo, devido a precisão do ponto flutuante da arquitetura de computador ou linguagem de programação usada para desenvolver o programa. Por tais motivos, foi adotada a análise do isomorfismo de subgrafos para os casos em que o isomorfismo do grafo como um todo não é alcançado.

A aplicação do isomorfismo de subgrafos tem o objetivo de verificar se uma determinada trilha de execução da reprodução é um da trilha da execução original, ou seja, é verificado se o grafo de proveniência da reprodução é um subgrafo do grafo de proveniência da execução. Um subgrafo S de G denotado por $S\{V, E\}$, possui algum subconjunto de vértices e arestas de V e E do grafo G, ou seja, S é subgrafo de G quando o grafo S for isomorfo a um subgrafo de G. Portanto, o isomorfismo faz a validação da reprodução dos ensaios de um experimento, permitindo avaliar se uma reprodução foi total ou parcialmente executada em relação ao ensaio original.

Equivalência nos Grafos de Proveniência

Cada *workflow* possui uma chave identificadora, e cada ensaio derivado desse *workflow* é identificado por essa chave identificadora concatenada ao número do ensaio.

Da mesma forma, uma atividade possui um identificador que é complementado pelos identificadores do *workflow* e do ensaio. As atividades possuem ainda as informações do tempo de início da execução, tempo de término, programas ou *scritps* vinculados e demais atributos. A função de análise da equivalência dos vértices da atividades $f : V_G \rightarrow V_H$ deve eliminar os identificadores do *workflow* e do ensaio e, em seguida, verificar se esse vértice analisado possui o mesmo identificador e demais atributos em ambos os grafos G e H.

A execução dos ensaios de um *workflow* sempre serão realizadas em um período de tempo t diferente. Um novo ensaio sempre é realizado em um período posterior. Por tal motivo, as análises de reprodutibilidade não levam em conta tipos de dados W3C Prov que são relacionados com atributos de tempo t. Por exemplo, para analisar se duas atividades possuem as mesmas características de metadados, tais como, identificador, *software* aplicativo relacionado ou parâmetros de configuração, é necessário excluir a análise de equivalência sobre o período em que essa atividade foi executada (tempo de início e tempo de fim). Isso se deve ao fato de que uma reprodução sempre será realizada em um momento diferente da execução original.

A exclusão dos elementos baseados em atributos de tempo t deve ser estendida para o caso das entidades arquivos que foram consumidas e produzidas pelos ensaios do *workflow*. Existem arquivos que incluem a data de acesso e o período em que o arquivo foi produzido. Esses atributos causam diferenças no conteúdo do arquivo em relação aos dados acessados e produzidos na reprodução considerando o ensaio original. Portanto, é necessário desconsiderar esses atributos durante a análise desses arquivos. Quando dois arquivos possuem exatamente o mesmo conteúdo é possível calcular a sua equivalência através do algoritmo MD5. Deve-se calcular o MD5 de cada um dos arquivos do experimento original e do experimento reproduzido e comparar os valores alcançados.

A função de equivalência dos vértices do tipo agente considera apenas o agente de *software*, que gerencia o *workflow*. Essa definição se deve pelo fato de que uma reprodução é desenvolvida para que terceiros possam reexecutar o experimento, de forma a avaliar o método e os resultados produzidos. Eventualmente, o próprio autor pode reproduzir o experimento que publicou, mas, em geral, a reprodução é feita por outros agentes. Diante disso, foi definido que o único agente a ser considerado na análise do isomorfismo é SGWfC responsável pela execução do *workflow*.

4.6.8 Gerando Artigo Executável

Ao final da implantação da pesquisa na nuvem, o servidor de aplicação aguarda as requisições de acesso aos objetos de pesquisa publicados ou a reprodução da pesquisa. O ReproeScience mantém um servidor de aplicação ativo para fazer a apre-

sentação da pesquisa reprodutível publicada pelo cientista. Esse servidor aguarda as requisições de acesso aos objetos de pesquisa compartilhados para que os metadados e dados de proveniência associados a ele possam ser analisados e obtidos por meio de *downloads*. O servidor de aplicação permite ainda a implantação e reprodução da pesquisa compartilhada dentro da infraestrutura de um provedor de nuvem, porém, usando as credenciais e recursos vinculados à conta do interessado na reprodução.

A Figura 4.14 apresenta o formulário de apresentação de uma atividade de um experimento implantado na nuvem sob o suporte do ReproeScience. Todas as informações apresentadas no formulário são obtidas a partir dos dados de proveniência do arquivo XML Prov que acompanha a publicação científica digital. O ReproeScience apresenta os dados de uma atividade do *workflow* assim que um usuário faz a solicitação, por meio do artigo científico digital. A Figura 4.15 mostra um exemplo de apresentação de uma entidade do tipo arquivo de dados.

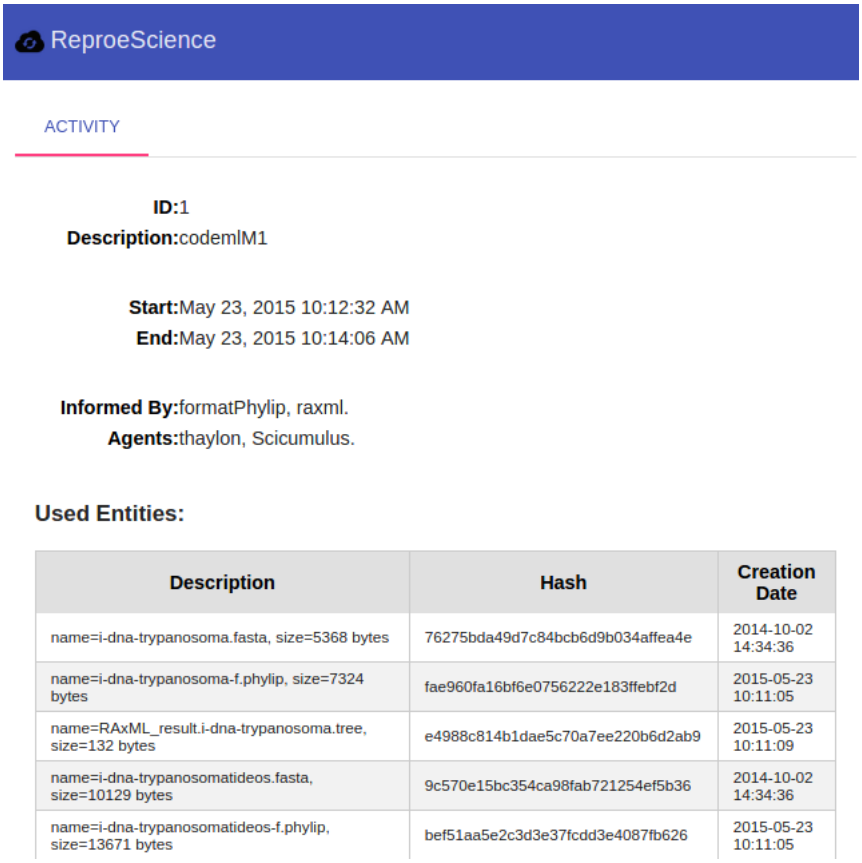


Figura 4.14: Apresentação dos metadados e dados de proveniência de uma atividade monitorada e referenciada pelo ReproeScience.

O ReproeScience dá suporte para a execução de um artigo científico digital vinculado aos objetos de pesquisa produzidos pelo *workflow* científico durante a execução do experimento. Ele possui mecanismos para que os objetos de pesquisa que foram compartilhados na nuvem possam ser acessados por meio de um código de referência, e permite que o experimento possa ser reimplantado e reproduzido, pois ele mantém

todos os objetos de pesquisa encapsulados em uma VM operacional (pronta para ser executada) armazenada em um provedor de computação em nuvem. Para reproduzir os elementos contidos em um artigo científico, basta o cientista acessar as VMs, através do RepreoScience, e comandar a sua implantação no provedor de nuvem suportado pelo ReproeScience.

Os objetos de pesquisa de um artigo são apresentados sob a forma de algoritmos, conjuntos de dados, figuras, gráficos, parâmetros de configuração e tabelas apresentados no corpo do artigo. Cada um desses elementos são associados com os agentes, as atividades e as entidades (conjuntos de dados, parâmetros, resultados finais e intermediários de um *workflow*) que compõem um experimento científico. A Figura 4.16 mostra como o ambiente operacional de um *workflow* científico fica encapsulado e preparado para executar os processos de reprodutibilidade.

Para que um artigo científico possa ter acesso à infraestrutura executável do ReproeScience é necessário incluir um conjunto de marcações no documento Latex em que o artigo será escrito, e ainda associar o artigo com a proveniência gerada para o experimento, através de um arquivo XML com a estrutura formada segundo os padrões do W3C Prov. No momento da produção do documento, o cientista deve incluir uma biblioteca de marcação do ReproeScience e utilizar os comandos para fazer as devidas referências aos agentes, atividades e entidades que compõem os dados de proveniência do experimento descrito no artigo. Essas referências são realizadas do documento Latex para o arquivo XML com as informações de proveniência. Cada uma dessas marcações faz referência a um ou mais objetos de

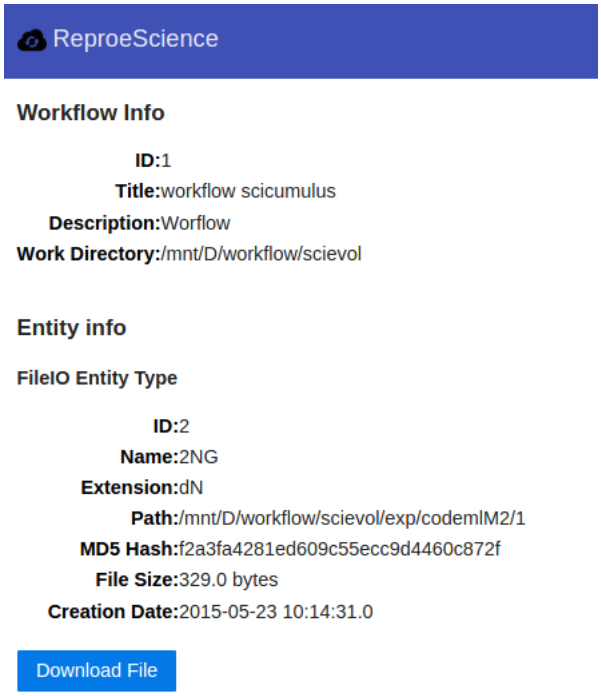


Figura 4.15: Apresentação dos metadados e proveniência de uma entidade.

pesquisa W3C PROV do *workflow* (agentes, atividades e entidades). Esse arquivo ".tex"é submetido ao módulo de compilação do ReproeScience para incluir o *link* do servidor de aplicação que possui a base de dados, e inserir o .xml do W3C Prov para converter as marcações. Neste *link* são inseridas as informações do id do *workflow*, ou seja, o identificador da entidade relacionada ao elemento, conforme mostrado na Figura 4.17. O código da esquerda apresenta o texto com um exemplo de anotação do Latex e o código da direita apresenta o resultado do processo de compilação. O arquivo XML contendo os dados de proveniência deve estar no mesmo diretório que o arquivo Latex,

Após o documento Latex ser compilado com as inserções das marcações, será gerado um arquivo PDF contendo as referências para a infraestrutura de execução do ReproeScience. As referências contidas no documento PDF são associadas com as entidades, agentes ou atividades contidas em um arquivo XML. Esse XML possui as informações de proveniência da execução do *workflow* usadas para a produção do conteúdo e resultados apresentados no documento PDF (o artigo científico propriamente dito). Essas informações são estruturadas segundo os padrões OPM e W3C Prov, e possuem anotações usadas pela abordagem ReproeScience para reproduzir

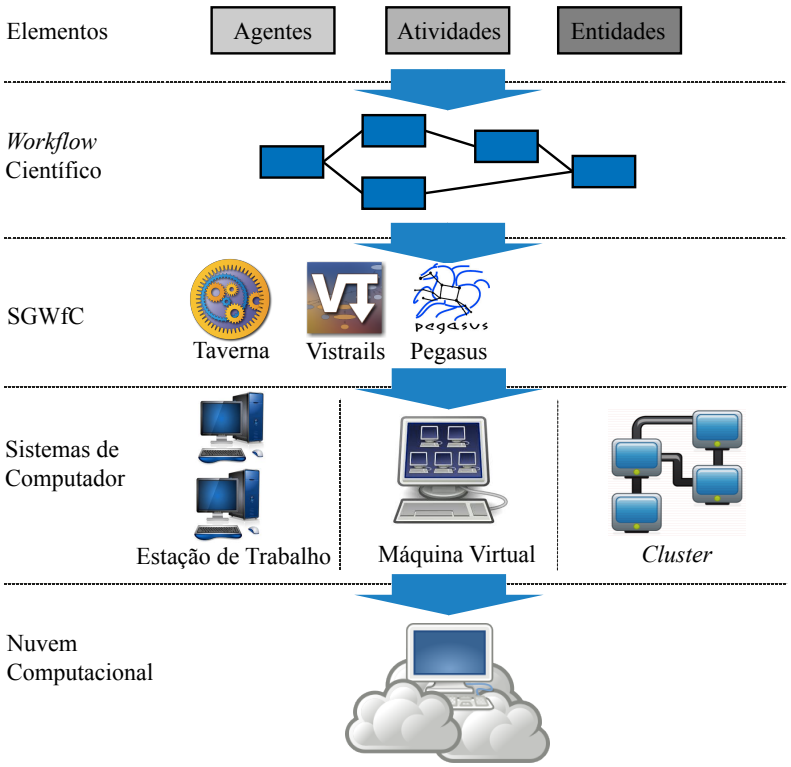


Figura 4.16: Esquema de encapsulamento dos elementos W3C Prov segundo a abordagem ReproeScience.

o *workflow*, e a sua infraestrutura de execução. Portanto, o arquivo XML que está associado ao artigo científico possui as informações necessárias para que o ReproeScience consiga recuperar a infraestrutura do experimento na nuvem e reproduzi-lo.

Conforme mencionado na Seção 4.4, o ReproeScience possui uma base de dados relacional contendo os dados de proveniência de um conjunto de experimentos sob a estrutura do modelo de dados da Figura 4.5. Esta base de dados possui informações complementares àquelas que estão armazenadas e descritas no arquivo XML que acompanha a publicação científica. O ReproeScience utiliza o arquivo XML para buscar e recuperar a infraestrutura necessária para reproduzir o *workflow*. Esse conjunto de informações permite que o componente possa contatar o provedor de nuvem que possui a infraestrutura do *workflow* e, em seguida, possa implantar o conjunto de VMs e demais elementos que formam a infraestrutura do experimento associado ao artigo científico. A Figura 4.18 mostra como é feito o acesso de um elemento de um experimento científico a partir de uma publicação.

É necessário que o código em linha de comando que solicita a execução do *workflow* científico esteja disponível e armazenado na base de dados, pois, as VMs na nuvem não podem ser acessadas através de interface gráfica. O acesso é feito através de linha de comando mediante o protocolo SSH, portanto, quando um leitor solicita a reprodução do artigo, o ReproeScience consulta o servidor *web* para recuperar as informações do experimento, em seguida, ele se comunica com a nuvem onde a infraestrutura do experimento está implantada, se necessário implanta as VMs (através do processo de instanciação), obtém o endereço das VMs implantadas e, ao final, através do comando de execução do experimento executa o *workflow* científico. Deve-se observar que essa nova execução produz um novo ensaio comandado por um novo agente, contendo novas entidades, ou seja, novos resultados intermediários e final. Esses novos elementos produzem uma nova assinatura *hash* dessa variação de execução do experimento mantendo a assinatura do experimento original integra, pois este último serviu apenas como um molde para a execução dos novos ensaios.

```
\documentclass{article}
\usepackage{graphicx}
\usepackage{hyperref}
\usepackage{embedfile}
\begin{document}

%ReproeScience("10")
\begin{figure}
    \centering
    \includegraphics[width=16cm]{img1.png}
    \caption{}
\end{figure}
%ReproeScienceEnd
A figura 1 representa uma entidade gerada pelo worflow.
\end{document}
```

```
\documentclass{article}

\usepackage{embedfile}
\usepackage{graphicx}
\usepackage{hyperref}
\begin{document}

\begin{figure}
    \centering
    \href{http://localhost:8080/Repro.html?idWorkflow=1&idEntity=10}
    {\includegraphics[width=16cm]{img1.png}}
    \caption{}
\end{figure}
A figura 1 representa uma entidade gerada pelo worflow.

\embedfile[ desc={XmlProvFile}]{Prov.xml}
\end{document}
```

Figura 4.17: Exemplo de marcação do documento Latex com referência a objetos de pesquisa.

O acesso aos agentes, atividades e entidades do arquivo segue uma estrutura hierárquica onde cada um dos elementos que compõe o experimento está encapsulado em outro. O ReproeScience identifica inicialmente qual provedor de nuvem armazena o experimento associado ao artigo científico. Esse processo é feito através do identificador do provedor de nuvem que fica associado aos dados de proveniência anexados ao artigo científico. Em seguida, o componente identifica quais são as VMs dentro do provedor de nuvem que compõem o conjunto de sistemas de computadores que formam a infraestrutura do experimento. Após identificadas essas VMs, o ReproeScience faz a sua devida implantação. O próximo nível da estrutura identifica qual o SWfMS que gerencia o *workflow* científico, bem como, todos os artefatos consumidos e produzidos pelo *workflow*. E, ao final, o ReproeScience permite que tanto o *workflow* quanto os seus elementos sejam referenciados pelo artigo científico. Portanto, o resultado final é a possibilidade de referencia e reprodução do conjunto de etapas e elementos utilizados na publicação de um artigo científico.

4.7 Considerações Finais

Este capítulo apresentou detalhadamente cada uma das etapas do desenvolvimento da abordagem de reprodutibilidade proposta nesta tese. Foram definidos cada um dos componentes, mostrando os detalhes da arquitetura interna e a interação com

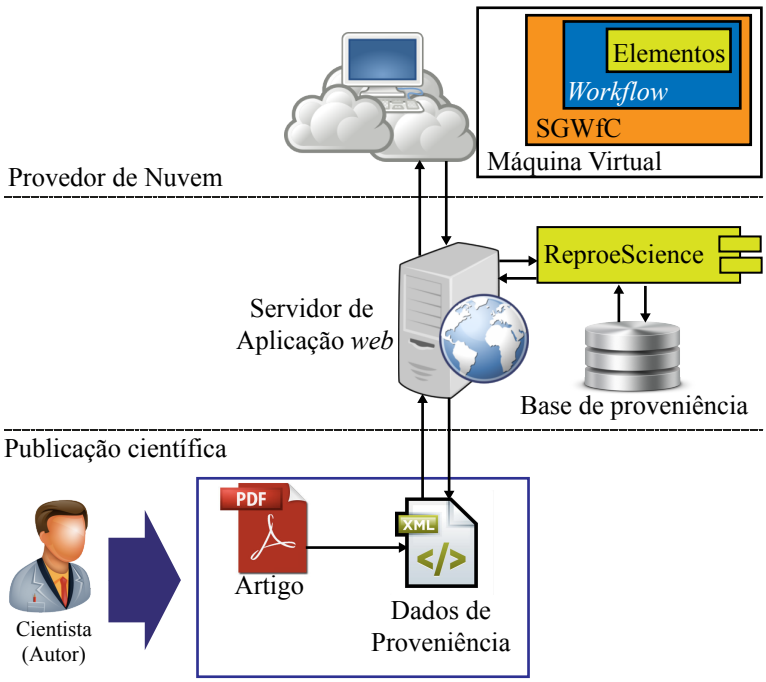


Figura 4.18: Etapas de acesso a infraestrutura de uma publicação científica reprodutível.

os elementos externos. Foi apresentado o modelo de dados e os procedimentos de armazenamento das características e comportamentos capturados durante a execução da reprodutibilidade, bem como os algoritmos empregados em cada uma das etapas do ciclo de vida do ReproeScience. Além disso, foi mostrado como um cientista pode utilizar a abordagem para a construção de um artigo científico reprodutível.

Os métodos e ferramentas definidos neste capítulo foram incorporados na execução de um conjunto de ensaios em experimentos modelados através de *workflows* científicos com o objetivo de avaliar a reprodutibilidade gerada através da abordagem proposta. Foram executados testes de *workflows* em ambientes sequenciais e paralelos e, o resultado do emprego da metodologia apresentada neste capítulo será discutido no Capítulo 5.

Capítulo 5

Avaliação do ReproeScience

Este capítulo apresenta uma avaliação dos resultados experimentais das reproduções dos *workflows* científicos selecionados para os estudos de caso com a aplicação da abordagem ReproeScience. Foram executados uma série de experimentos sob dois *workflows* implementados em duas áreas de pesquisa distintas, astronomia e bioinformática, analisando diferentes aspectos da reprodução dos experimentos. Em geral, foram analisados as características da reprodução dos resultados e da trilha de execução, esta última, a partir dos dados de proveniência. Foram avaliados os impactos do desempenho dos *workflows* primeiramente no ambiente de execução e, em seguida, nos dois ambientes selecionados para a reprodução com o objetivo de testar a metodologia proposta. Serão discutidos os aspectos como a transparência, portabilidade e cobertura de um experimento computacional reprodutível.

O capítulo está organizados em 5 seções. A Seção 5.1 apresenta a configuração dos ambientes com os recursos usados para a execução dos experimentos. A Seção 5.2 define os *workflows* científicos usados como estudos de caso para a execução dos ensaios, destacando as suas principais características e funções. A Seção 5.3 apresenta alguns exemplos de métodos de verificação dos resultados finais aplicados aos *workflows* selecionados. Além disso, discute o método para verificação e validação da trilha de execução da reprodução de um *workflow* em relação a um experimento original, baseado nos dados de proveniência. A Seção 5.4 apresenta a análise de desempenho da execução frente a adoção da abordagem de monitoramento para a reprodutibilidade. A Seção 5.7 apresenta as considerações finais sobre o capítulo.

5.1 Configuração do Ambiente do ReproeScience

Os *workflows* escolhidos para os experimentos são intensivos de dados e de computação e demandam por recursos de HPC para a execução e produção dos resultados das análises, portanto, foram implantados em uma infraestrutura de *cluster*, sob o sistema operacional Linux. O ambiente de execução usa um *cluster* do Núcleo de

Computação Aplicada da Universidade Federal do Rio de Janeiro (NACAD/COPPE/UFRJ) e foi implantado sob as infraestruturas de dois provedores de nuvem. Os recursos da infraestrutura da nuvem foram selecionados com base no perfil dos recursos computacionais do ambiente de execução, principalmente, em termos de instâncias com maior poder de processamento e memória principal. Foram selecionadas instâncias com o perfil de recursos mais similares a um menor custo monetário em dólares (US$). Foram selecionados dois provedores de nuvem para a realização dos testes, nomeados como Provedor A e Provedor B. A característica operacional dos ambientes são apresentadas na Tabela 5.1.

Tabela 5.1: Característica dos computadores usados nos experimentos.

Computer System	CPU	Cores/ CPU	RAM Memory	Operational System
Cluster local	Intel(R) Xeon(R) CPU X5650,@2.67GHz	8	2 Gb/Core	Suse Linux Enterprise Server 11
Provedor A	Intel Xeon E5-2666 v3 (Haswell) @2.9GHz	8	15 Gb/Core	Ubuntu Server 14.04
Provedor B	AMD Opteron(tm) Processor 4171 HE, @2.09 GHz	8	14 Gb/Core	Ubuntu Server 15.04

Os recursos de computação usados na execução original dos experimentos não são taxados para o usuários, no entanto, os recursos do provedor A foram obtidos sob o custo de $US\$0,46/hora$, e os custos relacionados aos recursos do provedor B foram equivalentes a $US\$0,72/hora$. Os ensaios realizados na avaliação da reprodução levaram em consideração as abordagens de monitoramento implementados com o componente *Ptrace*, usado em alguns trabalhos do estado da arte, e também pelo componente *inotify*, proposto neste trabalho. A implementação do *Ptrace* foi feita por meio de uma biblioteca denominada *Strace*[1]. A segunda implementação foi feita por meio do *inotify* e, para ambas as estratégias, foi necessário adaptar a execução do *Strace* para os ambientes de computação paralela e distribuída.

É importante destacar que tais ambientes fazem a troca de informações (mensagens, arquivos e etc) pela rede após um determinado número de núcleos, portanto, exigem algumas especificidades na sua implementação. Os computadores usados nos experimentos são limitados a 8 núcleos de processamento em cada nó, ou seja, a partir de 16 núcleos a implementação deve levar em consideração a comunicação via rede, uma vez que serão 2 nós. As execuções com 32 núcleos terão 4 nós com 8 núcleos cada, aumentando ainda mais a troca de mensagens na rede.

Esse cenário de computação paralela e distribuída apresentou a necessidade de implantação das VMs em ambientes com sistemas de arquivos distribuídos. Os experimentos mostraram que apenas a implantação dos dados, sistema operacional, *software*, sistema de *workflow*/*workflows* e demais elementos não são suficientes

[1]http://linux.die.net/man/1/strace

para garantir a execução da reprodução de um experimento neste tipo de ambiente. Além deste conjunto de elementos é preciso identificar as especificidades de um ambiente especial e prover os mecanismos para que ele funcione adequadamente e, nesse caso, é preciso implantar um sistema de arquivos distribuídos nas VMs para que a reprodução ocorre de forma correta e tenha um desempenho adequado.

É importante observar que outras estratégias de monitoramento que capturam chamadas de sistema e interrupções precisam estar implantadas no mesmo nível que os *hypervisors*, com acesso direto ao *hardware* usado para a execução. Em muitos casos, a exemplo dos casos usados neste trabalho, o cientista não tem acesso a este nível e geralmente não possui poderes administrativos para gerenciar o ambiente/sistema de computação no qual a pesquisa é desenvolvida. Portanto, a implementação deste cenário foi desconsiderada neste trabalho, pois traz um desafio operacional para a configuração e implantação dos experimentos.

5.2 *Workflows* dos Estudos de Caso

Os experimentos foram realizados com a execução de dois *workflows* científicos distintos: (1) o SciEvol, da biologia computacional, e o (2) Montage, da área da astronomia. Ambos *workflows* demandam pelo emprego de recursos especiais para a execução das suas atividades. Entretanto, cada um possui um conjunto de características distintas que criam a oportunidade de testar a reprodução em diferentes contextos de experimentos científicos em larga escala. Em geral, o SciEvol demanda por maior recursos de processamento, principalmente nas atividades de análise de evolução. Já o Montage é um *workflow* mais intensivo de dados, produzindo uma grande quantidade de arquivos com grande volume de dados. Por exemplo, o SciEvol gera uma quantidade de dados de aproximadamente $100Mbyte$ por execução, enquanto no Montage, um conjunto de arquivo de entrada reduzido possui pelo menos 4 arquivos com mais de $100Mbyte$ cada um e, ao final, com uma entrada reduzida gera mais de $15Gbytes$ de dados por execução.

Os *workflows* projetados no Vistrails e SciCumulus permitem analisar os contextos de execução em em estações de trabalho, ambiente de *clusters* e provedores de nuvem. O *workflow* SciEvol é aplicado a uma das áreas de maior estudo na biologia computacional, a análise filogenética [97]. Já o *workflow* Montage foi usado em diversos trabalhos na literatura para a análise da capacidade e comportamentos de execução dos provedores de nuvem [16, 98, 99].

5.2.1 *Workflow* Científico SciEvol

Muitos estudos evolucionários são baseados em simulações de computador, relacionando vários programas aplicados a bioinformática para a construção de métodos de Reconstrução da Evolução Molecular (MER) [7]. Os experimentos MER focam na inferência de relações evolutivas entre indivíduos, populações, espécies e entidades taxonômicas, mas utilizando um conjunto de dados moleculares. O gerenciamento dos experimentos MER envolve o processamento de um grande conjunto de dados em escala de milhares de genomas, sendo um problema intensivo de dados devido a sua característica exploratória. As execuções do MER exploram diversos valores de parâmetros e diferentes modelos evolutivos de substituição de códons.

O SciEvol é capaz de inferir a evolução e analisar as forças e mecanismos dos processos evolucionários, avaliando todas as variações possíveis dos códons [7]. O modelo de substituição de códons do SciEvol utiliza o programa codeml, disponível no pacote PAML [**?**]. Conforme mostrado na Figura 5.1, o SciEvol implantado no SGWfC SciCumulus é composto por 13 atividades [7]:

1. Executa um *script* na linguagem python para formatar o conjunto de dados de entrada (retira os códons de parada);

2. Constrói os alinhamentos de múltiplas sequências (MSA) usando o programa MAFFT. Essa etapa recebe um arquivo multi-fasta pré formatado como entrada e, ao final, produz o alinhamento MSA através de um arquivo de saída no formato FASTA;

3. É feita a execução do programa RefSeq para converter um alinhamento MSA em FASTA para o tipo de arquivo PHYLIP. Os arquivos PHYLIP são usados para a construção de árvores filogenéticas. Nessa etapa são gerados os arquivos PHYLIP denominados phylip-file-one;

4. Essa etapa recebe os arquivos PHYLIP do tipo phylip-file-one e os converte para um segundo formato de arquivos PHYLIP, denominado phylip-file-two. Essa conversão é necessária, pois, esse segundo formato é usado na atividade de análise evolucionária. Nesse ponto, as sequências são organizadas de forma que sejam múltiplas de três (ajuste necessário para se usar o programa codeml);

5. São construídas as árvores filogenéticas a partir do programa RAxML. Esta etapa recebe um arquivo phylip-file-one como entrada e, ao final, produz uma árvore filogenética como saída;

6. Essa etapa encapsula 6 sub atividades que representam a execução das fases da exploração MER através do Codeml. A entrada dessa etapa possui 3

parâmetros: as árvores filogenéticas produzidas na etapa anterior, os arquivos phylip-file-one e o arquivo de controle denominado codeml.ctl. A saída dessa atividade produz um conjunto de arquivos com informações evolucionárias;

7. A última etapa processa os arquivos de saída da execução das etapas anteriores para obter os resultados estatísticos e representar os locais sob os modelos de códons evolucionários no gráfico.

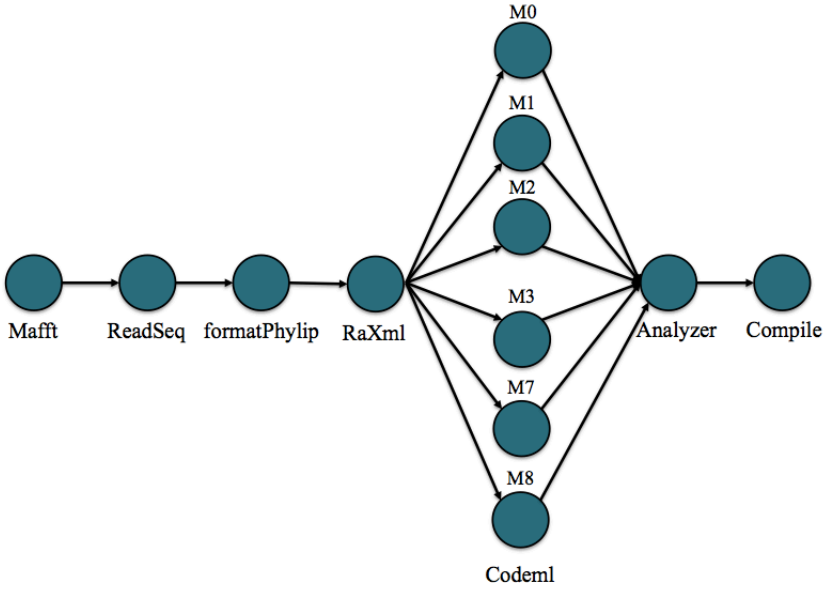


Figura 5.1: *Workflow* Científico SciEvol (Adaptado de [7]).

O SciEvol executa uma pesquisa exploratória que demanda por técnicas de computação paralela e ambientes especializados de HPC para obter os resultados em tempo computacionalmente hábil. Diante dessas características, o SciEvol foi desenvolvido para ser nativamente executado com o SGWfC SciCumulus para explorar os benefícios da estratégia de computação HPC no ambiente de nuvem. O SciEvol foi implantado no SGWfC Vistrails com fins de avaliação da abordagem de reprodutibilidade sob um cenário diferente.

5.2.2 *Workflow* Científico Montage

O Montage é um kit de ferramentas usado para montar imagens astronômicas do tipo FITS (*Flexible Image Transport System*) em mosaicos personalizados [12] [16]. Este formato é adotado pela comunidade astronômica para o armazenamento e compartilhamento de dados de arquivos. Os dados armazenados neste tipo de arquivo registram a relação das coordenadas dos pixels da imagem e as unidades físicas, baseados no Sistema Mundial de Coordenadas (*World Coordinate System* - WCS). O WCS inclui uma definição de como as coordenadas e as projeções celestes são representados no formato FITS [12]. Ele analisa os valores contidos no arquivo FITS

para identificar as regiões das imagens no céu e, em seguida, calcula o mosaico de imagens relativos a essa região descrita pelos valores contidos no arquivo.

A entrada de dados do Montage é composta pela região no céu em que se deseja iniciar a composição do mosaico, o tamanho do mosaico em termos de grau quadrado e outros parâmetros, tais como o arquivo da imagem. Ao final da sua execução, ele produz um arquivo FITS compatível com os parâmetros de especificação da imagem definidos no arquivo de entrada com o mosaico de saída reprojetado com as coordenadas do espaço informadas na entrada de dados [90]. O Montage fornece um utilitário que permite a apresentação do conteúdo do arquivo FITS em uma imagem no formato *Joint Photographic Experts Group* (JPEG). O valor científico do Montage é derivado de três características gerais do seu projeto [12]:

- Ele usa algoritmos que preservam a calibração e a fidelidade posicional (astrométrica) das imagens de entrada para entregar mosaicos que atendam aos parâmetros de projeção, coordenadas e escala espacial. Ele suporta todos os sistemas de coordenadas e projeções em uso na astronomia.

- Ele contém módulos independentes para analisar a geometria das imagens no céu, e para a criação e gerência dos mosaicos. Esses módulos são ferramentas poderosas em seu próprio domínio e tem aplicabilidade fora da produção de mosaico, em áreas como a validação de dados.

- Ele foi escrito em um compilador ANSI (*American National Standards Institute*) C, é portátil e escalável. Pode ser executado em *desktops*, *clusters* ou ambientes de supercomputadores baseados em sistemas Unix.

Tabela 5.2: Características científicas do Montage [12].

Característica	**Descrição**
Precisão	Preserva a fidelidade espacial e calibração de imagens de entrada.
Portabilidade	Funciona em todas as plataformas Linux/Unix.
Escalabilidade	Executa em ambientes *desktops*, *clusters* e grades computacionais
Disponibilidade	O código fonte e documentação do usuário são abertos e disponíveis para *download*.
Generabilidade	Suporta toda as projeções dos Sistemas de Coordenadas Mundial (WCS) e sistemas de coordenadas comuns.
Desempenho	Processa 40 milhões de *pixels* em até 32 minutos a 128 nós em um *cluster* Linux.
Flexibilidade	Possui motores independentes para analisar a geometria das imagens no céu, reprojetando imagens; retificando as emissões do plano de fundo a um nível comum, co-adicionando imagens.
Conveniência	Possui ferramentas para o gerenciamento e manipulação de grandes arquivos de imagens.

O Montage produz os mosaicos das imagens em quatro etapas distintas. Cada uma das etapas é independente da outra, possibilitando a inclusão ou a variação do

método aplicado em alguma dessas etapas. O Montage permite ainda que o usuário explore técnicas de paralelização, através do uso de bibliotecas MPI (*Message Passing Interface*) ou com um framework para o mapeamento de *workflow* científico dentro de uma grade computacional, criando um plano de execução no SGWfC Pegasus [2]. A avaliação do Montage foi realizada com o suporte do SGWfC SciCumulus aplicando o MPI. As etapas de produção do mosaico no Montage são apresentadas nos itens abaixo:[12]:

1. Descobrir a geometria das imagens de entrada no céu a partir das palavras-chave no arquivo FITS, de forma a usá-la para calcular a geometria da saída do mosaico no céu;

2. Reprojetar imagens de entrada para que tenham a mesma escala espacial, o mesmo sistema de projeção WCS e rotação de imagem de coordenadas equivalentes;

3. Modelar a radiação do plano de fundo das imagens de entrada para atingir fluxo de escalas e níveis de plano de fundo comuns nos mosaicos;

4. Co-adicionar as imagens, com correção de fundo reprojetada em um mosaico.

O *workflow* Montage foi modelado no SGWfC SciCumulus em um total de nove atividades: (1) ListFits é a etapa inicial que obtém um arquivo comprimido e faz a sua devida extração para descompactar os arquivos FITS de entrada, (2) Projection filtra um conjunto de arquivos FITS a partir das palavras, (3) SelectProjections, (4) CreateUncorrentedMosaic, (5) CalculeOverlaps, (6) ExtractDifferences, (7) CalculateDifference, (8) FitPlane, e (9) CreateMosaic. As execuções e reproduções do Montage foram avaliadas sob o gerenciamento do SGWfC SciCumulus.

5.3 Análise da Equivalência

Existem três formas de verificar a equivalência das execuções originais em relação a suas posteriores reproduções. A primeira forma pode ser feita por meio da verificação visual dos dados derivados, analisando se os áudios, gráficos, imagens e tabelas representam os resultados finais da execução de um *workflow*, ou seja, se eles são equivalentes. A segunda forma envolve verificar se os resultados obtidos, ou seja, o conjunto de dados de saída são equivalentes nas duas execuções, comparando se os valores numéricos alcançados atendem a uma escala de equivalência.A terceira forma, proposta nesta tese, inclui a análise da taxa de reprodução da trilha de execução para a geração dos resultados da execução original em relação a reprodução.

[2]http://pegasus.isi.edu/

Neste caso, são avaliados todos os tipos de dados (conjunto de dados de entrada, derivados/intermediários, finais, metadados e proveniência) usados e gerados pelo experimento para verificar o quanto os dados usados e gerados em todo o processo são equivalentes.

5.3.1 Análise Visual de Equivalência dos Dados Derivados

A análise visual de equivalência permite verificar os dados derivados gerados após a execução dos experimentos. Os dados derivados são uma representação mais conveniente dos resultados. Eles são resultado da transformação dos dados analíticos para uma representação que usa recursos multimídia como gráficos, imagens e tabelas para enriquecer a apresentação dos resultados. Por exemplo, os resultados finais do *workflow* SciEvol são representados pelos dados e informações de um conjunto de matrizes evolucionárias contidas em arquivos texto. Trata-se de um conjunto de valores numéricos que representam o resultado de uma análise evolucionária. Para melhorar a interpretação dos resultados, as informações são apresentadas por meio de representações visuais mais convenientes, facilitando a análise. conforme os gráficos mostrados na Figura 5.2.

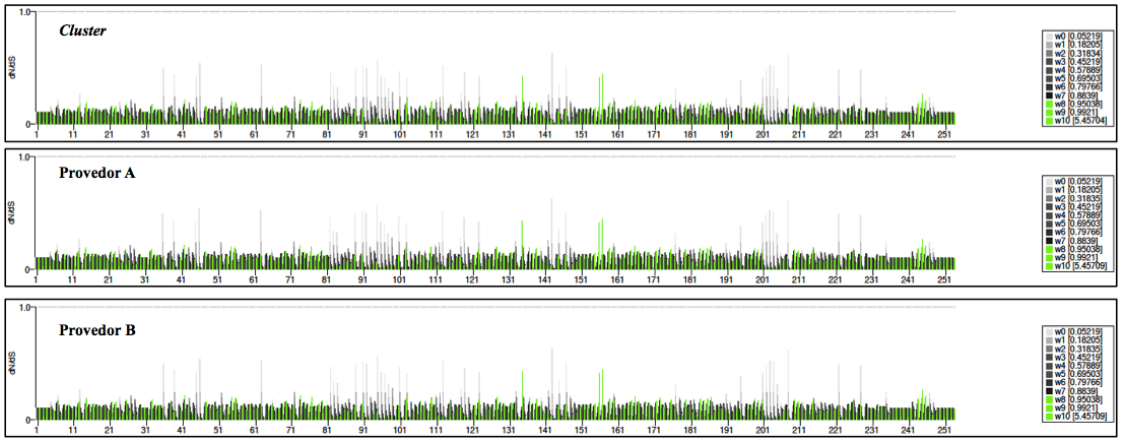


Figura 5.2: Verificação visual dos gráficos de análises evolucionárias geradas pela execução do SciEvol no *cluster* (primeiro gráfico da imagem), e posteriores reproduções no provedor A (segundo gráfico) e provedor B (terceiro gráfico).

Os gráficos foram gerados, respectivamente, a partir da execução e reprodução do SciEvol. Estes gráficos mostram o resultado da análise evolucionária gerada pelo *workflow* científico apresentado na Figura 5.1. A primeira imagem foi obtida a partir de uma execução do SciEvol no *cluster*, e a segunda e terceira imagens foram geradas com a reprodução do experimento nos provedores de nuvem. Visualmente as reproduções apresentadas na parte inferior da imagem (Provedor A e B) geraram resultados bastante similares em relação com a execução original, apresentada no gráfico na parte superior da imagem (*cluster*). Portanto, nesse método de análise

da reprodução, é preciso que o cientista avalie visualmente os dados derivados do resultado para verificar se duas execuções de um mesmo *workflow* resultaram na reprodução do resultado.

O mesmo tipo de análise pode ser aplicada nos dados derivados gerados nos resultados da execução do *workflow* Montage. Os resultados da execução do Montage são apresentados em uma imagem com um mosaico de uma determinada região do espaço, obtida a partir de um determinado conjunto de coordenadas armazenadas nos arquivos binários no formato FITS. A Figura 5.3 apresenta o mosaico gerado pela execução (imagem da direita), e o mosaico gerado com a posterior reprodução (imagem da esquerda) do Montage. É possível verificar que trata-se de imagens bastante similares, inclusive nas partes que apresentam um tom de cinza mais acentuado.

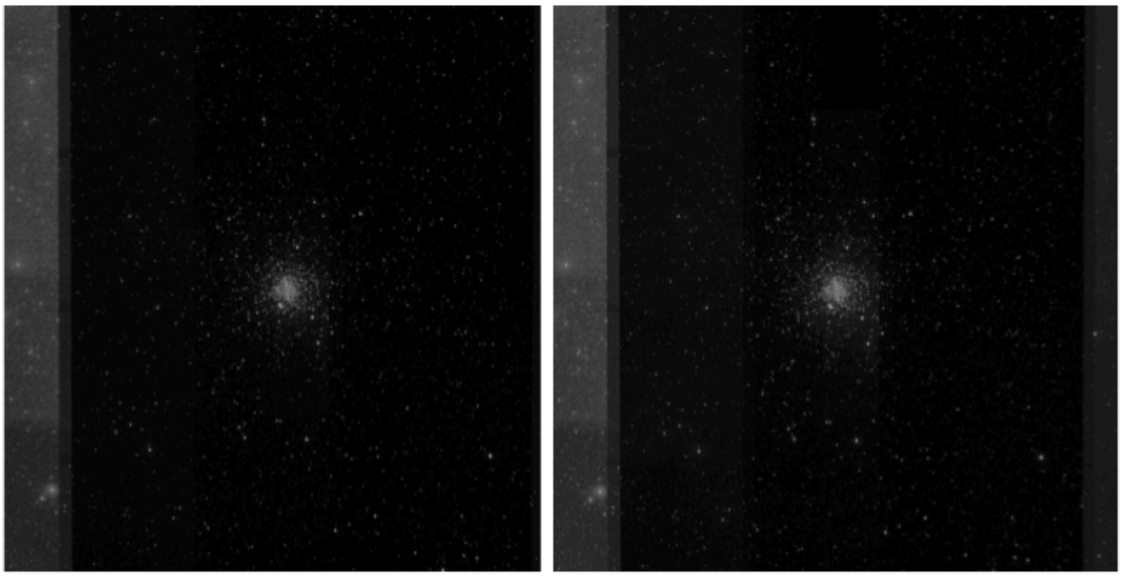


Figura 5.3: Verificação visual dos mosaicos de imagens do espaço obtidos com a execução (a esquerda) e reprodução (a direita) do Montage.

A verificação visual é a forma de análise padrão adotada por todas as abordagens de reprodução descritas no estado da arte. As execuções e reproduções de ambos *workflows* dos estudos de caso geraram resultados com total equivalência visual, independente do ambiente usado (*cluster*, provedor A e provedor B), ou característica dos recursos computacionais, tais como o número de núcleos usados na execução ou o tipo de instância de VM selecionada.

5.3.2 Análise da Equivalência dos Resultados

Essa análise é feita por meio da verificação da precisão numérica dos dados gerados como resultados finais. Essa análise é executada comparando analiticamente os valores obtidos no experimento original em relação aos alcançados pela reprodução.

Para fazer esta análise é necessário identificar os conjuntos de dados que representam o resultado final, na execução original e na posterior reprodução e, em seguida, extrair os percentuais de equivalência na comparação de ambos os valores. A partir desses percentuais foi possível verificar a precisão obtida nos resultados em ambas execuções. Os gráficos mostrados na Figura 5.2 foram derivados dos arquivos apresentados na Figura 5.4. A análise de equivalência apresentada na Seção 5.3.1 torna mais conveniente a apresentação e conferência das análises, porém, não avalia os resultados em termos analíticos.

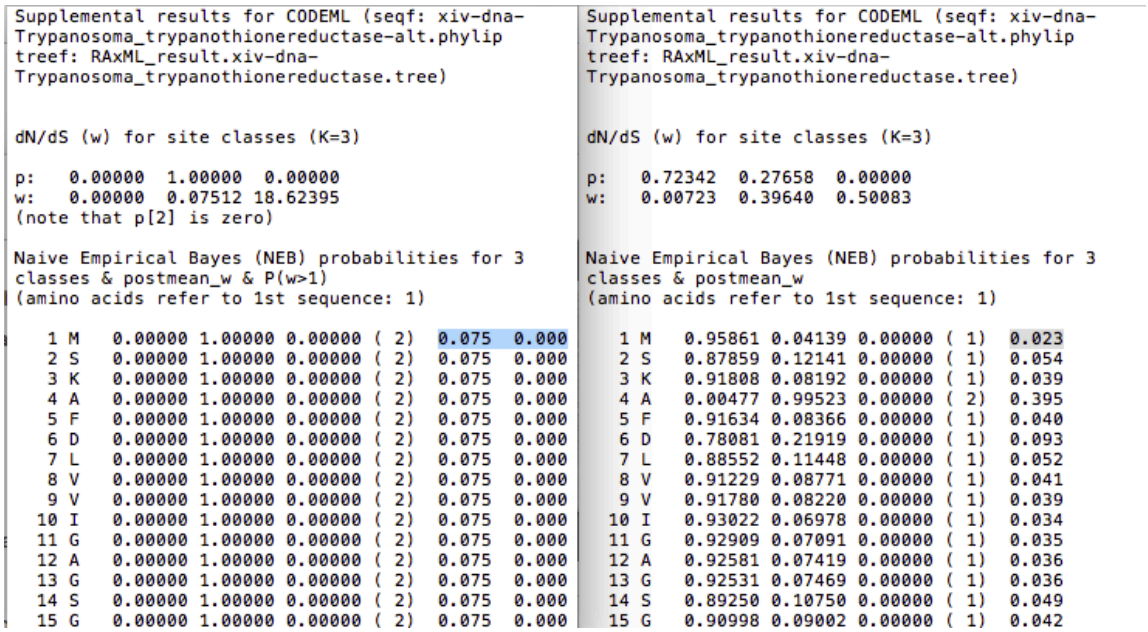
A Figura 5.4 apresenta o arquivo com o resultado final gerado pelo SciEvol. Este arquivo foi produzido pela atividade *Codeml* do *workflow* apresentado na Figura 5.1. Ele possui uma matriz com um conjunto de valores que representam as informações evolucionárias resultantes dos processos de análise executados por todas as atividades anteriores. A análise de equivalência dos resultados para o caso específico do SciEvol deve ser aplicada a esse conjunto de valores. A verificação da precisão dos resultados neste caso é feita por meio da comparação dos valores das duas últimas colunas desta matriz (marcadas em azul) na figura. São comparados o arquivo de resultado da reprodução em relação ao arquivo de resultado da execução que originou os resultados publicados. Neste caso, essa comparação foi calculada avaliando a proporção percentual dos valores, ou seja, foi verificada a porcentagem de equivalência dos valores reproduzidos em relação aos da execução original.

```
Supplemental results for CODEML (seqf: xiv-dna-Naegleriagruberi_thioredoxinreductase-alt.phylip  treef:
RAxML_result.xiv-dna-Naegleriagruberi_thioredoxinreductase.tree)


dN/dS (w) for site classes (K=11)

p:   0.08969  0.08969  0.08969  0.08969  0.08969  0.08969  0.08969  0.08969  0.08969  0.08969  0.10307
w:   0.23492  0.66078  0.87702  0.96286  0.99095  0.99835  0.99980  0.99999  1.00000  1.00000  6.67513

Naive Empirical Bayes (NEB) probabilities for 11 classes & postmean_w & P(w>1)
(amino acids refer to 1st sequence: 1)

   1 *   0.50178 0.11540 0.06201 0.04938 0.04594 0.04508 0.04491 0.04489 0.04489 0.04489 0.00084 ( 1)  0.572  0.001
   2 V   0.03334 0.10394 0.10928 0.10745 0.10651 0.10625 0.10619 0.10618 0.10618 0.10618 0.00849 ( 3)  0.969  0.008
   3 G   0.00615 0.05707 0.08854 0.10017 0.10382 0.10476 0.10495 0.10497 0.10497 0.10497 0.11963 (11)  1.639  0.120
   4 T   0.09847 0.11821 0.10454 0.09819 0.09609 0.09554 0.09543 0.09542 0.09541 0.09541 0.00729 ( 2)  0.908  0.007
   5 Y   0.08333 0.11228 0.10477 0.10043 0.09891 0.09851 0.09843 0.09842 0.09842 0.09842 0.00808 ( 2)  0.926  0.008
   6 -   0.14697 0.10247 0.08986 0.08602 0.08488 0.08459 0.08454 0.08453 0.08453 0.08453 0.06707 ( 1)  1.218  0.067
   7 -   0.08969 0.08969 0.08969 0.08969 0.08969 0.08969 0.08969 0.08969 0.08969 0.08969 0.10307 (11)  1.471  0.103
   8 -   0.09365 0.10895 0.09903 0.09475 0.09336 0.09300 0.09293 0.09292 0.09292 0.09292 0.04556 ( 2)  1.133  0.046
   9 *   0.02402 0.08378 0.10073 0.10506 0.10623 0.10652 0.10658 0.10659 0.10659 0.10659 0.04732 (10)  1.204  0.047
  10 Q   0.16382 0.13767 0.10197 0.08934 0.08546 0.08446 0.08427 0.08424 0.08424 0.08424 0.00029 ( 1)  0.813  0.000
  11 E   0.70349 0.09266 0.03730 0.02664 0.02394 0.02328 0.02315 0.02314 0.02314 0.02314 0.00012 ( 1)  0.425  0.000
  12 -   0.05451 0.08766 0.09255 0.09359 0.09385 0.09391 0.09393 0.09393 0.09393 0.09393 0.10822 (11)  1.527  0.108
  13 -   0.08969 0.08969 0.08969 0.08969 0.08969 0.08969 0.08969 0.08969 0.08969 0.08969 0.10307 (11)  1.471  0.103
  14 -   0.08969 0.08969 0.08969 0.08969 0.08969 0.08969 0.08969 0.08969 0.08969 0.08969 0.10307 (11)  1.471  0.103
  15 -   0.08969 0.08969 0.08969 0.08969 0.08969 0.08969 0.08969 0.08969 0.08969 0.08969 0.10307 (11)  1.471  0.103
  16 -   0.08969 0.08969 0.08969 0.08969 0.08969 0.08969 0.08969 0.08969 0.08969 0.08969 0.10307 (11)  1.471  0.103
  17 -   0.08969 0.08969 0.08969 0.08969 0.08969 0.08969 0.08969 0.08969 0.08969 0.08969 0.10307 (11)  1.471  0.103
  18 -   0.08969 0.08969 0.08969 0.08969 0.08969 0.08969 0.08969 0.08969 0.08969 0.08969 0.10307 (11)  1.471  0.103
  19 -   0.08969 0.08969 0.08969 0.08969 0.08969 0.08969 0.08969 0.08969 0.08969 0.08969 0.10307 (11)  1.471  0.103
  20 *   0.10530 0.09730 0.09377 0.09245 0.09203 0.09192 0.09190 0.09189 0.09189 0.09189 0.05965 ( 1)  1.209  0.060
```

Figura 5.4: Arquivos com equivalência em 100% dos resultados produzidos da reprodução em relação à execução.

A análise da equivalência dos resultados deve identificar a proporção de similaridade entre cada arquivo de dados comparado. Devem ser analisados os pares de arquivos correspondentes da reprodução em relação à execução original. A Figura

5.5 apresenta um par de arquivos correspondentes cuja similaridade foi de 45%, ou seja, bastante baixa em termos de reprodução. Essas diferenças ocorrem devido a divergência na precisão dos cálculos, porém, mesmo ocorrendo essa baixa proporção de equivalência entre um outro par de arquivos de dados correspondentes, a análise deve ser aplicada a todo o conjunto de dados avaliado. Devem-se obter todos os arquivos gerados na reprodução e na execução, identificar os pares correspondentes, calcular a equivalência de cada par e, ao final, verificar a média do conjunto como um todo.



```
Supplemental results for CODEML (seqf: xiv-dna-
Trypanosoma_trypanothionereductase-alt.phylip
treef: RAxML_result.xiv-dna-
Trypanosoma_trypanothionereductase.tree)


dN/dS (w) for site classes (K=3)

p:   0.00000  1.00000  0.00000
w:   0.00000  0.07512 18.62395
(note that p[2] is zero)

Naive Empirical Bayes (NEB) probabilities for 3
classes & postmean_w & P(w>1)
(amino acids refer to 1st sequence: 1)

   1 M   0.00000 1.00000 0.00000 ( 2)  0.075  0.000
   2 S   0.00000 1.00000 0.00000 ( 2)  0.075  0.000
   3 K   0.00000 1.00000 0.00000 ( 2)  0.075  0.000
   4 A   0.00000 1.00000 0.00000 ( 2)  0.075  0.000
   5 F   0.00000 1.00000 0.00000 ( 2)  0.075  0.000
   6 D   0.00000 1.00000 0.00000 ( 2)  0.075  0.000
   7 L   0.00000 1.00000 0.00000 ( 2)  0.075  0.000
   8 V   0.00000 1.00000 0.00000 ( 2)  0.075  0.000
   9 V   0.00000 1.00000 0.00000 ( 2)  0.075  0.000
  10 I   0.00000 1.00000 0.00000 ( 2)  0.075  0.000
  11 G   0.00000 1.00000 0.00000 ( 2)  0.075  0.000
  12 A   0.00000 1.00000 0.00000 ( 2)  0.075  0.000
  13 G   0.00000 1.00000 0.00000 ( 2)  0.075  0.000
  14 S   0.00000 1.00000 0.00000 ( 2)  0.075  0.000
  15 G   0.00000 1.00000 0.00000 ( 2)  0.075  0.000
```

```
Supplemental results for CODEML (seqf: xiv-dna-
Trypanosoma_trypanothionereductase-alt.phylip
treef: RAxML_result.xiv-dna-
Trypanosoma_trypanothionereductase.tree)


dN/dS (w) for site classes (K=3)

p:   0.72342  0.27658  0.00000
w:   0.00723  0.39640  0.50083


Naive Empirical Bayes (NEB) probabilities for 3
classes & postmean_w
(amino acids refer to 1st sequence: 1)

   1 M   0.95861 0.04139 0.00000 ( 1)  0.023
   2 S   0.87859 0.12141 0.00000 ( 1)  0.054
   3 K   0.91808 0.08192 0.00000 ( 1)  0.039
   4 A   0.00477 0.99523 0.00000 ( 2)  0.395
   5 F   0.91634 0.08366 0.00000 ( 1)  0.040
   6 D   0.78081 0.21919 0.00000 ( 1)  0.093
   7 L   0.88552 0.11448 0.00000 ( 1)  0.052
   8 V   0.91229 0.08771 0.00000 ( 1)  0.041
   9 V   0.91780 0.08220 0.00000 ( 1)  0.039
  10 I   0.93022 0.06978 0.00000 ( 1)  0.034
  11 G   0.92909 0.07091 0.00000 ( 1)  0.035
  12 A   0.92581 0.07419 0.00000 ( 1)  0.036
  13 G   0.92531 0.07469 0.00000 ( 1)  0.036
  14 S   0.89250 0.10750 0.00000 ( 1)  0.049
  15 G   0.90998 0.09002 0.00000 ( 1)  0.042
```

Figura 5.5: Arquivos com equivalência em 45% dos resultados produzidos da reprodução em relação a execução.

É importante observar que cada tipo de *workflow* possui formas particulares de análise dos resultados. Essa característica limita a busca por métodos de avaliação genéricos com estratégias comuns ou padronizadas para avaliar os resultados da aplicação do *workflows*. Portanto, para fazer a avaliação analítica dos resultados produzidos pelos experimentos reproduzidos em relação aos originalmente consolidados em uma publicação, é necessário que cada cientista crie uma atividade de avaliação (testes) para gerar uma descrição padrão na forma de metadados para que as comparações seja feitas sob esses metadados.

5.3.3 Análise da Equivalência da Proveniência

A análise da equivalência da proveniência valida o fluxo de execução da reprodução em relação a execução do experimento original. São avaliados os estados e comportamentos de cada um os elementos de proveniência que compõem ambos os experimentos. Porém, a abordagem de reprodução proposta foi implantada em ambientes

com experimentos com características orientadas a dados e, nesse sentido, a avaliação da equivalência da proveniência baseada em grafos isomórficos é aplicada ao comportamento de uso e geração do conjunto de arquivos gerados pelos *workflows*.

Foi observado nos estudos de casos que a geração de um arquivo em uma determinada atividade é baseado no conteúdo de arquivos previamente usados, ou seja, um arquivo lido na entrada de dados de uma atividade, unido com os parâmetros de configuração, determina a necessidade de criação de um novo arquivo, ou ainda, qual será o conteúdo de um arquivo gerado como resultado da execução dessa atividade.

A análise da equivalência da proveniência deve levar em consideração as características do uso, produção e conteúdo dos arquivos manipulados na execução de um *workflow* de um experimento computacional. Portanto, as análises de proveniência propostas com a abordagem ReproeScience são aplicadas na avaliação da descrição e do conteúdo desses arquivos. Foi observado que os nomes dos arquivos em grande parte dos experimentos possuem uma nomenclatura padrão, muitas vezes orientadas pelo conteúdo dos dados usados como entrada de dados da atividade corrente. O conteúdo dos arquivos foi identificado por meio de funções de *hash* que geram uma codificação referente aos dados e informações armazenadas no conteúdo do arquivo.

Os *workflows* foram analisados sob a perspectiva de três conjuntos de equivalências baseados em uma operação de conjunção: (1) Arquivos que possuem apenas os nomes equivalentes, (2) Arquivos que possuem apenas o valor do *hash* equivalente e (3) arquivos que possuem o nome e também o valor de *hash* iguais, conforme mostrado na Figura 5.6. Foi observado em alguns experimentos que características da arquitetura do sistema de computação podem interferir na geração do conteúdo desse arquivo, por exemplo, precisão da máquina em operações de ponto flutuante, porém, não é um fator determinante para grandes alterações nos estados e comportamentos dos elementos usados e produzidos por um *workflow*.

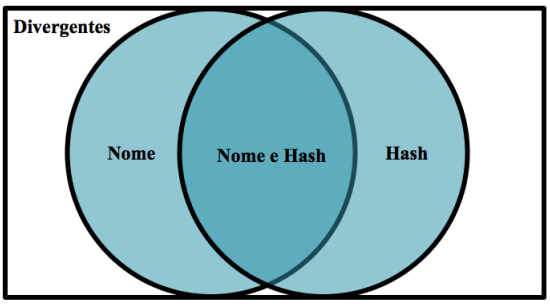


Figura 5.6: Conjuntos formados pelos atributos analisados nome e código *hash*.

Este conjunto de arquivos usados e gerados pelos *workflows* são representados por vértices nos grafos de proveniência das execuções e reproduções no ReproeScience. Os dados e informações da descrição e conteúdo dos arquivos são os elementos usados na análise do isomorfismo das trilhas de execução. Em um primeiro momento, o grafo

de proveniência da reprodução é comparado ao grafo de proveniência da execução para verificar se ele é um subgrafo, bem como a sua proporção de equivalência. Em uma segunda análise, é verificado se a morfologia do grafo de reprodução é equivalente à morfologia do grafo de execução, por meio do algoritmo de isomorfismo VF2. Essa análise determina que a forma do grafo devem ser equivalente apesar do conteúdo das arestas divergirem em algum ponto. Em uma última análise é verificado se ocorreu uma equivalência total, ou seja, se os grafos são isomorfos.

Produção dos Elementos W3C Prov - *Workflow* SciEvol

Foram realizadas análises sob o conjunto de elementos W3C Prov usados e produzidos durante a execução do SciEvol, usando os SGWfCs SciCumulus e Vistrails. O quantitativo de elementos W3C Prov gerado pelas execuções são apresentados na Tabela 5.3. Nesta tabela são mostradas as execuções monitoradas pelas estratégias *Strace* e *inotify*, essa última estratégia denominada *Repro*, a partir desse ponto. São apresentados 4 diferentes cenários de execução, o primeiro usando o SGWfC Vistrails com 8 núcleos, e os demais, no SciCumulus com 8, 16 e 32 núcleos. O objetivo dos testes em diferentes cenários é a verificação da robustez do método de monitoramento independente das alterações de tecnologia de SGWfC.

Analisando os dados na tabela é possível verificar que o quantitativo de elementos foram bastante similares tanto nas execuções no *cluster* quanto nos provedores de nuvem, independente da estratégia de monitoramento adotada, *Strace* ou *Repro*. Podem ser observadas algumas diferenças na produção de valores na execução de 16 cores com o monitor *Strace* (em negrito). Porém, grande parte das execuções do SciEvol resultou grafos isomórficos segundo o algoritmo VF2. O quantitativo de atividades e agentes, assim como os relacionamento de atribuição e delegação foram omitidos na tabela pelo fato de serem geradas instâncias e valores constantes desses elementos nas execuções. Foram definidos 2 agentes no SciEvol, sendo eles o usuário do sistema operacional e o próprio SGWfC. Nesse caso, o primeiro agente delega a execução do *workflow* para o segundo. O *workflow* foi projetado no Vistrails com 6 atividades e no SciCumulus com 7 atividades.

Uma das execuções do SciEvol gerou uma estrutura de grafos apresentada na Figura 5.7. O grafo representa as interações de uso (arestas em azul) e geração (arestas em vermelho) de entidades pelas 6 atividades do SciEvol no Vistrails. Os vértices que apresentam pontos de concentração entre as arestas vermelhas e azuis representam as atividades do *workflow*. Os vértices de união entre as arestas azuis e vermelhas são as entidades geradas por uma atividade antecessora consumida por uma atividade predecessora. Cada um dos nós e arcos do grafo possui um conjunto de metadados e dados de proveniência, conforme mostrado na Figura 5.8. Essas informações são extraídos para uma representação em arquivos XML sob o modelo

Tabela 5.3: Dados da produção de elementos W3C Prov da execução (*cluster*) e reprodução (Provedores A e B) do SciEvol.

8 Núcleos - Vistrails						
W3C Prov	**Cluster**		**Provedor A**		**Provedor B**	
Elements	**Strace**	**Repro**	**Strace**	**Repro**	**Strace**	**Repro**
Entities	4.558	4.558	4.558	4.558	4.558	4.558
Used	1.358	1.358	1.358	1.358	1.358	1.358
Generated	4.006	4.006	4.006	4.006	4.006	4.006
Informed	801	801	801	801	801	801
Attributed	4.006	4.006	4.006	4.006	4.006	4.006
8 Núcleos - SciCumulus						
W3C Prov	**Cluster**		**Provedor A**		**Provedor B**	
Elements	**Strace**	**Repro**	**Strace**	**Repro**	**Strace**	**Repro**
Entities	6.187	6.187	6.187	6.187	6.187	6.187
Used	2.392	2.392	2.392	2.392	2.392	2.392
Generated	5.756	5.756	5.756	5.756	5.756	5.756
Informed	1.450	1.450	1.450	1.450	1.450	1.450
Attributed	5.756	5.756	5.756	5.756	5.756	5.756
16 Núcleos - SciCumulus						
W3C Prov	**Cluster**		**Provedor A**		**Provedor B**	
Elements	**Strace**	**Repro**	**Strace**	**Repro**	**Strace**	**Repro**
Entities	**6.156**	6.188	**6.188**	6.188	**6.188**	6.188
Used	2.404	2.404	2.404	2.404	2.404	2.404
Generated	**5.716**	5.756	**5.756**	5.756	**5.756**	5.756
Informed	**1.442**	1.450	**1.450**	1.450	**1.450**	1.450
Attributed	**5.716**	5.756	**5.756**	5.756	**5.756**	5.756
32 Núcleos - SciCumulus						
W3C Prov	**Cluster**		**Provedor A**		**Provedor B**	
Elements	**Strace**	**Repro**	**Strace**	**Repro**	**Strace**	**Repro**
Entities	6.190	6.190	6.190	6.190	6.190	6.190
Used	2.428	2.428	2.428	2.428	2.428	2.428
Generated	5.756	5.756	5.756	5.756	5.756	5.756
Informed	1.450	1.450	1.450	1.450	1.450	1.450
Attributed	5.756	5.756	5.756	5.756	5.756	5.756

W3C PROV que serão mostrados a seguir.

Os trechos de arquivos XML mostrados a seguir foram gerados a partir das informações de proveniência das execuções do *workflow* Scievol. São apresentadas as representações de uma *entity*, uma *activity*, um relacionamento *used* e um relacionamento *wasGeneratedBy* do W3C PROV. As *tags* iniciadas com "*res*"definem as anotações dos metadados e dados de proveniência gerados pelos atributos descritos no modelo de dados do ReproeScience. Todos os elementos monitorados pelo ReproeScience devem ter informações de identificação e código *hash*.

O *hash* codifica o conteúdo do arquivo e auxilia na verificação da consistência das entidades. Ele permite analisar se o objeto de pesquisa usado ou gerado em um experimento possui o conteúdo equivalente ao que foi implantado e publicado

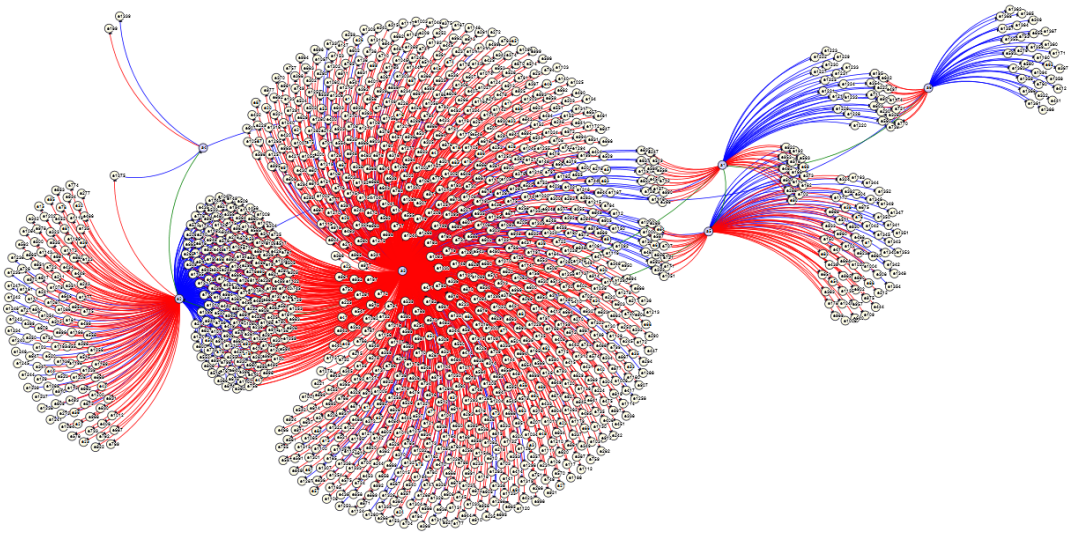


Figura 5.7: Grafo de proveniência produzido com a execução do *workflow* SciEvol apoiado pelo SGWfC Vistrails.

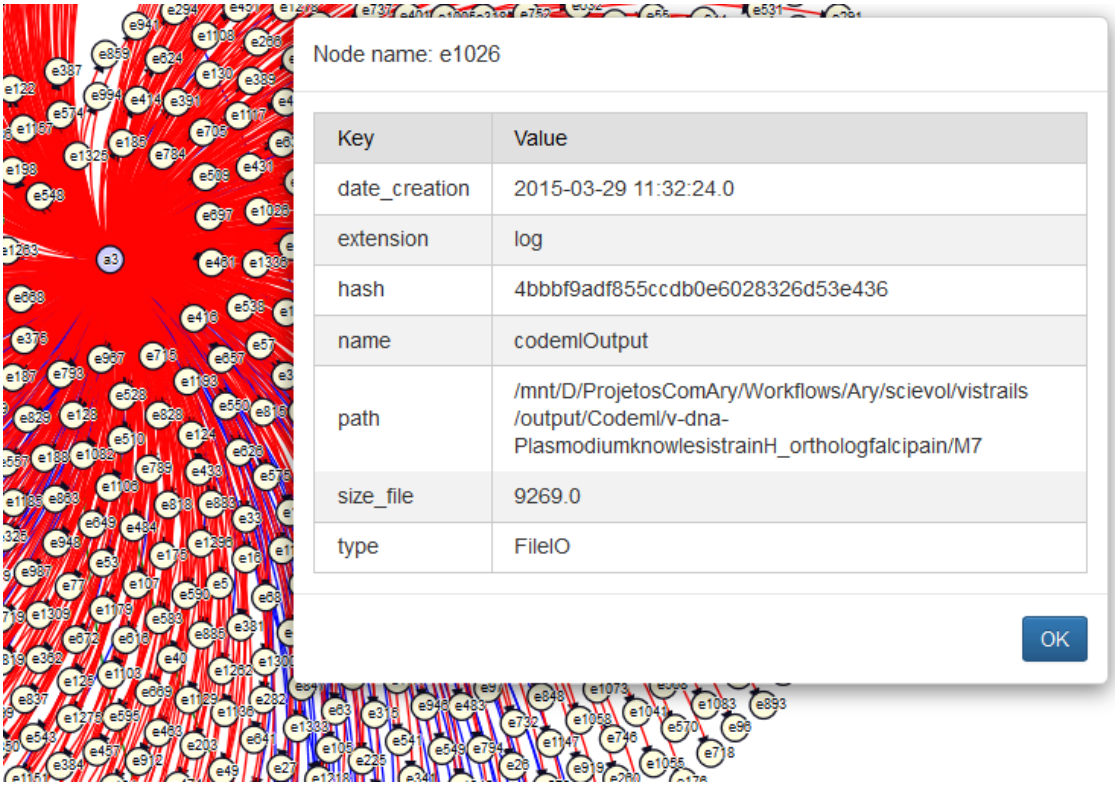


Figura 5.8: Informações de metadados e dados de proveniência contidas nos grafos de proveniência.

na nuvem. O código *hash* também é usado nas tarefas de validação da reprodução da pesquisa reprodutível para verificar se os conteúdos gerados são equivalentes na execução, e na sua posterior reprodução.

No caso das entidades, é necessário identificar o tipo de entidade que está sendo tratada. Existe um conjunto de atributos genéricos para as entidades e cada tipo

possui um conjunto de atributos específicos. Atualmente, o ReproeScience monitora o uso e geração de três diferentes entidades: (1) arquivos - FileIO, (2) parâmetros - Parameter e (3) sistemas de computadores - Computer System. O exemplo apresentado a seguir representa uma entidade arquivo. Nesse caso são identificados os dados sobre o nome, extensão e caminho com a localização do arquivo no sistema de arquivos, um identificador, o valor do *hash*, tamanho do arquivo e a data de criação, assim como o tipo, nesse caso "FileIO".

```
<prov:entity prov:id="e1117">
        <res:name> codemlOutput </res:name>
        <res:extension>log</res:extension>
        <res:path>
                /mnt/D/.../i-dna-trypanosomatideos/M3
        </res:path>
        <res:hash>
                ef6442953ca9c9faf6e3a5eee965a99d
        </res:hash>
        <res:size_file>11295.0</res:size_file>
        <res:date_creation>
                2015-03-29_11:21:10.0
        </res:date_creation>
        <res:type>FileIO</res:type>
</prov:entity>
```

A entidade a seguir representa um computador usado na execução de um experimento. Trata-se de um exemplo do tipo de entidade ComputerSystem. Essa entidade armazena todas as informações operacionais desse computador com metadados e dados de proveniência sobre as características de rede, sistema operacional, capacidade de armazenamento primário e secundário, processador. Esta entidade relaciona-se com as atividades apenas no relacionamento *used*, onde um computador é usado para a execução de uma atividade. As informações sobre essa entidade auxiliam na obtenção dos recursos nos provedores de nuvem, orientando sob quais infraestruturas computacionais as atividades de um experimento foram submetidas.

```
<prov:entity prov:id="e1218">
    <res:version>3.13.0-37-generic</res:version>
    <res:external_network>100</res:external_network>
    <res:internal_network>100</res:internal_network>
    <res:number_processors>8</res:number_processors>
    <res:processor_speed>800.0</res:processor_speed>
    <res:hard_disc>1.94109239296E11</res:hard_disc>
    <res:ram_memory>6.0996608E9</res:ram_memory>
    <res:computer>null</res:computer>
```

```
    <res:ip>192.168.1.107</res:ip>
    <res:mac>68-A3-C4-8F-1B-82</res:mac>
    <res:type>ComputingSystem</res:type>
    <res:SO_name>Linux</res:SO_name>
    <res:SO_version>3.13.0-37-generic</res:SO_version>
    <res:SO_plataform>amd64</res:SO_plataform>
  </prov:entity>
```

A atividade, mostrada a seguir, representa um processo de execução de um programa descrito como ReadSeq do *workflow* SciEvol, e está associado a um dado comando de execução. Este processo é iniciado e finalizado em um determinado momento por um agente identificado pelo código 2 na execução de número 0 do *workflow* científico de código 1. O ReproeScience distingue o número da execução, permitindo que a verificação e validação da proveniência possa ser feita sob uma ou mais execuções. Qualquer execução posterior do *workflow* é considerada uma reprodução segundo as definições da abordagem. Pode ser observado que existem três definições "prov"padrão, uma para a identificação da atividade e duas para a definição do momento de início e finalização da sua execução, conforme a notação do W3C PROV,

```
<prov:activity prov:id="a1">
        <prov:startTime>
                2015-03-24_09:46:48.579
        </prov:startTime>
        <prov:endTime>
                2015-03-24_09:50:08.105
        </prov:endTime>
        <res:execution>0</res:execution>
        <res:description>ReadSeq</res:description>
        <res:state>null</res:state>
        <res:command>
                java -jar %=WFDIR%/bin/readseq.jar
                -all -f=12 %=NAME%.mafft -o %=NAME%.phylip
        </res:command>
        <res:workflow>1</res:workflow>
        <res:agent>2</res:agent>
</prov:activity>
```

Os dois trechos de definições XML a seguir representam os relacionamentos de uso e geração de entidades. Eles são responsáveis pelas definições das arestas dos grafos de análise de proveniência propostos na abordagem ReproeScience.

```
<prov:used prov:id="4">
        <prov:activity prov:ref="a1"/>
```

```
        <prov:entity prov:ref="e354"/>
</prov:used>

<prov:wasGeneratedBy prov:id="16">
        <prov:entity prov:ref="e90"/>
        <prov:activity prov:ref="a1"/>
</prov:wasGeneratedBy>
```

O *workflow* SciEvol foi projetado de forma diferente nos SGWfCs SciCumulus e Vistrails, baseado na estratégia de execução específica de cada sistema. O principal objetivo é verificar a reprodução dos metadados e proveniênica de um mesmo *workflow* em duas tecnologias de SGWfC distintas. Mesmo alterando o número de atividades e a estratégia de execução do mesmo *workflow* é preciso verificar por meio do monitoramento se a abordagem consegue atingir a reprodução na mesma proporção.

Conforme mencionado anteriormente, no Vistrails foram definidas 6 atividades (Mafft, ReadSeq, RaXml, Codeml, Analyzer e Compiler). Nesse caso, a atividade *Codeml* é tratada como uma atividade individual. No SciCumulus foi incluída a atividade *formatPhylip* entre as atividades *RealdSeq* e *RaXml*. Essa inclusão de atividade aumentou o quantitativo de entidades do tipo arquivo. As atividades do SciEvol no SciCumulus foram projetadas para serem executadas de forma paralela e individualizada, ou seja, cada variação da atividade *Codeml* (que possui 6 variações) foi projetada como uma atividade específica e, portanto, uma execução independente de outra com uma variação diferente do *Codeml*.

As execuções dos *workflows* nos diferentes SGWfC geraram diferentes usos e produções de entidades, isso se deve ao fato de que cada SGWfC possui sua própria estratégia de gerenciamento do *workflow*. Portanto, observando as Figuras 5.9 e 5.10 fica clara a diferença de entidades produzidas pelo SciEvol no Vistrails e no SciCumulus, vindo o primeiro caso a criar aproximadamente 30% de entidades e relacionamentos a mais que o segundo caso, o que já era esperado.

Ao analisar as execuções (*cluster*) e reproduções (provedor A e B) do SciEvol em cada uma das estratégias, SGWfC Vistrails e SciCumulus, é possível observar que a produção dos elementos W3C Prov são bastante próximas em todas os gráficos (Figuras 5.9, 5.10, 5.11 e 5.12). Portanto, são criados elementos W3C PROV na execução original e reprodução na mesma proporção. Além disso, pode ser observado que as execuções no SGWfC SciCumulus nos ambientes de 8, 16 e 32 núcleos possuem as mesmas características de produção de elementos W3C Prov. Se forem comparados os resultados obtidos *inter* núcleos será observado que todas as execuções e reproduções produziram entidades *FileIO* de forma equivalente.

Por exemplo, se uma execução for realizada em um ambiente de 8 núcleos e, posteriormente, reproduzida em um provedor de nuvem usando 16 núcleos de pro-

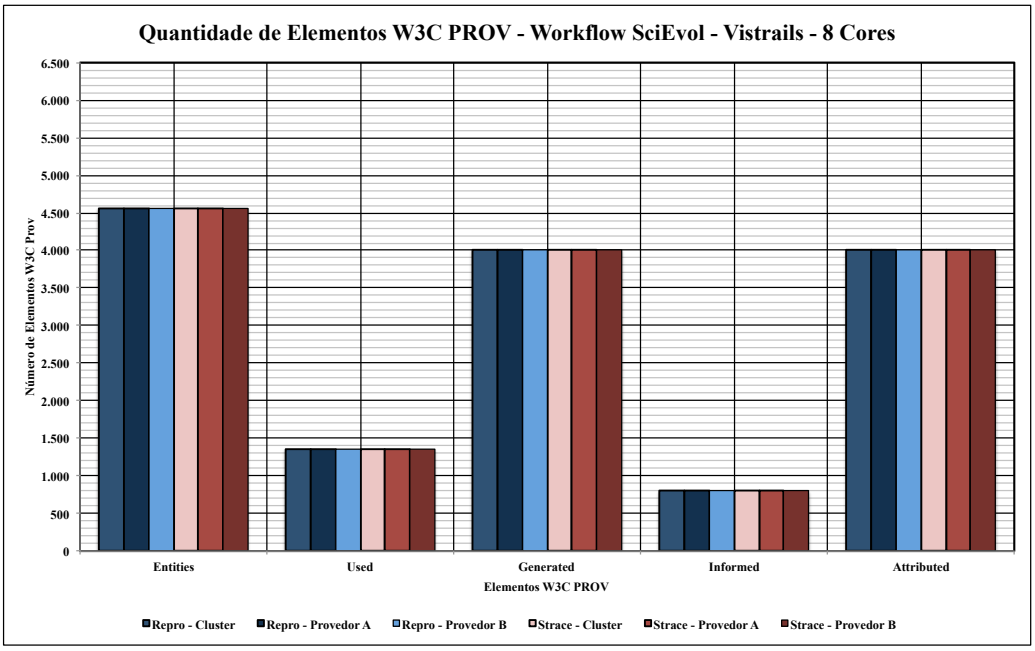


Figura 5.9: Equivalência dos arquivos de dados da execução e reprodução do SciEvol apoiado pelo SGWfC Vistrails usando 8 núcleos de processamento.

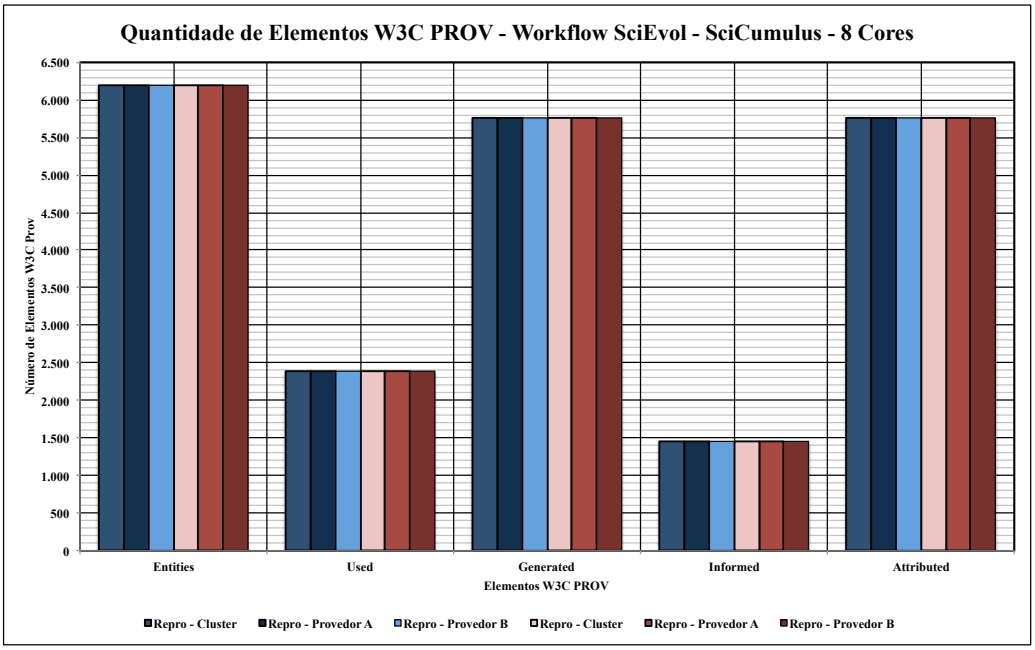


Figura 5.10: Equivalência dos arquivos de dados da execução e reprodução do SciEvol apoiado pelo SGWfC SciCumulus usando 8 núcleos de processamento.

cessamento, a verificação da reprodução das entidades do tipo arquivo será realizada em um primeiro momento sob arquivos de dados como um todo, considerando que todos foram realizados pelo *workflow*, e não por cada nó de computação. Em seguida, a análise leva em conta o que cada nó produziu, porém, como informação complementar e não como restritiva. Serão produzidos o mesmo número de entidades do tipo arquivo, e o ReproeScience será capaz de verificar que a reprodução

ocorreu independente da estratégia de alocação de recursos adotada. Entretanto, deve-se observar que algumas características de proveniência serão alteradas, por exemplo, o número de entidades do tipo "Sistema de Computador"e, portanto, vão gerar um quantitativo de elementos W3C Prov diferentes.

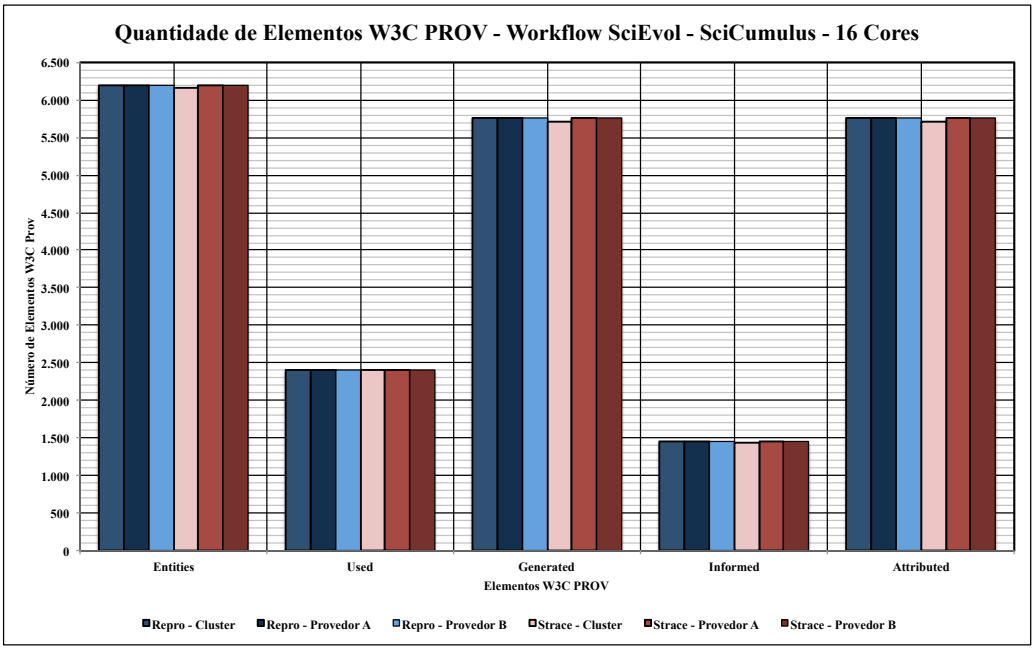


Figura 5.11: Equivalência dos arquivos de dados da execução e reprodução do SciEvol apoiado pelo SGWfC SciCumulus usando 16 núcleos de processamento.

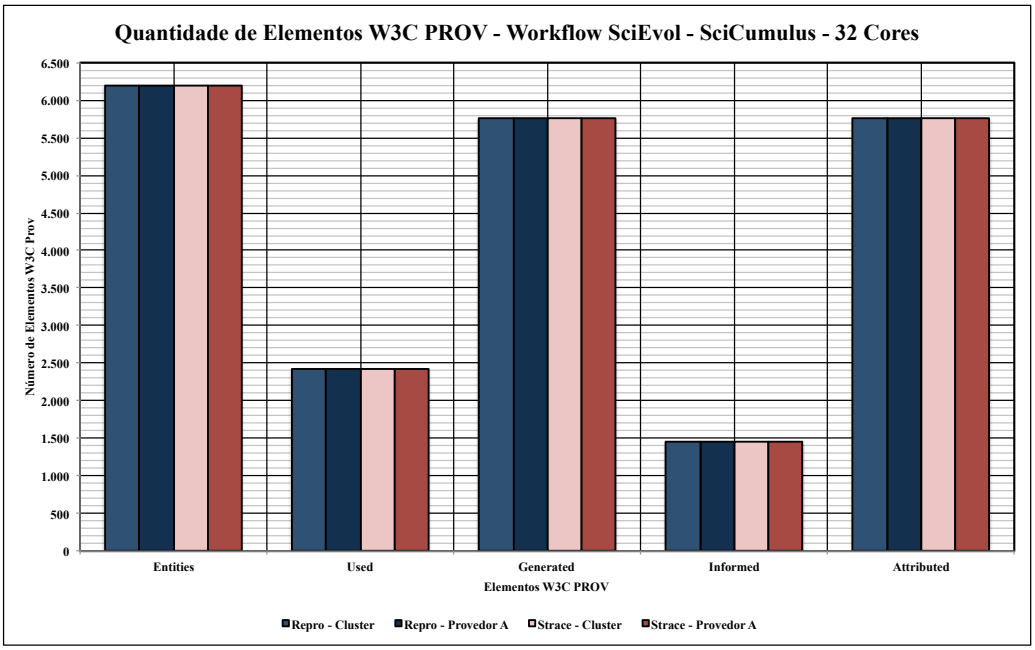


Figura 5.12: Equivalência dos arquivos de dados da execução e reprodução do SciEvol apoiado pelo SGWfC SciCumulus usando 32 núcleos de processamento.

Análise da Equivalência da Proveniência - *Workflow* SciEvol

A análise da equivalência da proveniência avaliou os arquivos de dados produzidos segundo os critérios discutidos na Seção 5.3.3. Foram extraídas as informações das equivalências do nome e dos valores de *hash* dos arquivos de dados manipulados pelas duas versões do *workflow* SciEvol. O SciEvol teve uma versão implantada inicialmente no SGWfC Vistrails e, em seguida, no SGWfC SciCumulus. O principal objetivo foi observar a taxa de uso e geração dos arquivos nas duas diferentes abordagens de gerenciamento de *workflow*.

Os dados sobre a equivalência da proveniência das execuções são apresentados na tabela 5.4. Esta tabela apresenta os dados classificados por estratégia de monitoramento e os ambientes que foram comparados. Os ambientes colocados a esquerda do símbolo "x"são os ambientes de execução, já os que estão a direita do símbolo são os ambientes de reprodução. Foram separados todos os arquivos válidos produzidos pelos *workflows*. Os arquivos válidos são aqueles que possuem informações de entrada de dados e dados produzidos pelas atividades do *workflow*. Os dados gerados pelos SGWfC para a gestão do *workflow* foram desconsiderados. A tabela mostra três colunas contento as informações de equivalência: (1) nome, (2) *hash* e (3) nome e *hash*. A coluna quantidade de equivalência soma o valor das 3 equivalências e baseado no total de arquivos válidos é extraída a taxa de reprodução do *workflow*.

Tabela 5.4: Taxa de reprodução de arquivos do *Workflow* SciEvol com Vistrails e SciCumulus.

8 Cores - Vistrails								
Estratégia	**Ambiente Original x Reproduzido**	**Arquivos Válidos (Original)**	**Arquivos Válidos (Reproduzido)**	**Nome Equivalente**	**Hash Equivalente**	**Nome/Hash Equivalentes**	**Quantidade de Equivalência**	**Taxa de Reprodução (%)**
Repro	**Cluster x Provedor A**	4.006	4.006	2.156	7	1.841	4.004	99,95
	Cluster x Provedor B	4.006	4.006	2.159	8	1.838	4.005	99,98
	Provedor A x Provedor B	4.006	4.006	1.973	7	2.024	4.004	99,95
Strace	**Cluster x Provedor A**	4.006	4.006	2.128	13	1.863	4.004	99,95
	Cluster x Provedor B	4.006	4.006	2.135	7	1.861	4.003	99,93
	Provedor A x Provedor B	4.006	4.006	2.027	7	1.970	4.004	99,95
8 Cores - SciCumulus								
Estratégia	**Ambiente Original x Reproduzido**	**Arquivos Válidos (Original)**	**Arquivos Válidos (Reproduzido)**	**Nome Equivalente**	**Hash Equivalente**	**Nome/Hash Equivalentes**	**Quantidade de Equivalência**	**Taxa de Reprodução (%)**
Repro	**Cluster x Provedor A**	3.862	3.862	1.675	0	2.187	3.862	100,00
	Cluster x Provedor B	3.862	3.862	1.704	0	2.158	3.862	100,00
	Provedor A x Provedor B	3.862	3.862	801	0	3.061	3.862	100,00
Strace	**Cluster x Provedor A**	3.862	3.862	1.678	1	2.182	3.861	99,97
	Cluster x Provedor B	3.862	3.862	1.700	1	2.160	3.861	99,97
	Provedor A x Provedor B	3.862	3.862	794	2	3.064	3.860	99.95
16 Cores - SciCumulus								
Estratégia	**Ambiente Original x Reproduzido**	**Arquivos Válidos (Original)**	**Arquivos Válidos (Reproduzido)**	**Nome Equivalente**	**Hash Equivalente**	**Nome/Hash Equivalentes**	**Quantidade de Equivalência**	**Taxa de Reprodução (%)**
Repro	**Cluster x Provedor A**	3.862	3.862	1.655	6	2.197	3.858	99,95
	Cluster x Provedor B	3.862	3.862	1.672	5	2.180	3.857	99,87
	Provedor A x Provedor B	3.862	3.862	737	18	3.102	3.857	99,87
Strace	**Cluster x Provedor A**	3.854	3.862	1.678	0	2.176	3.854	99,79
	Cluster x Provedor B	3.854	3.862	1.681	0	2.173	3.854	99,79
	Provedor A x Provedor B	3.862	3.862	732	30	3.094	3.856	99.84
32 Cores - SciCumulus								
Estratégia	**Ambiente Original x Reproduzido**	**Arquivos Válidos (Original)**	**Arquivos Válidos (Reproduzido)**	**Nome Equivalente**	**Hash Equivalente**	**Nome/Hash Equivalentes**	**Quantidade de Equivalência**	**Taxa de Reprodução (%)**
Repro	**Cluster x Provedor A**	3.862	3.862	1.660	9	2.187	3.856	100,00
	Cluster x Provedor B	3.862	3.862	1.653	8	2.195	3.856	99,84
	Provedor A x Provedor B	3.862	3.862	728	24	3.106	3.856	99,90
Strace	**Cluster x Provedor A**	3.862	3.862	1.673	4	2.183	3.856	99,95
	Cluster x Provedor B	3.862	3.862	1.677	4	2.177	3.856	99,90
	Provedor A x Provedor B	3.862	3.862	735	21	3.100	3.856	99,84

A execução dos *workflows* mostrou que a quantidade de arquivos usados e gerados varia na casa de algumas dezenas comparando o uso e produção de arquivos na reprodução e na execução. O SciEvol manipulou uma média de 4.000 arquivos com uma taxa média de reprodução acima de 99% usando as estratégias *Strace* e *Repro*, enquanto no SciCumulus foi obtida uma média de 3.850 arquivos manipulados a uma taxa de 100% com a abordagem *Repro* e acima de $99,95\%$ com o Strace. As Figuras 5.13 e 5.14 mostram, respectivamente, a quantidade de equivalência dos arquivos usados e gerados nos experimentos originais e reproduzidos.

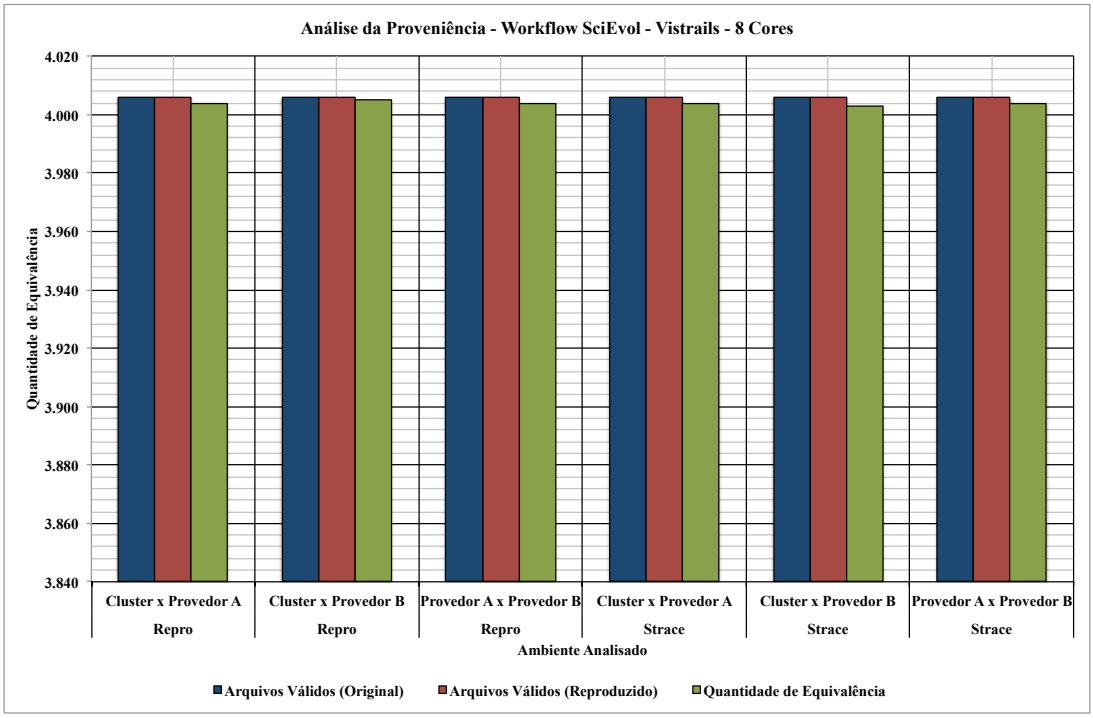


Figura 5.13: Equivalência dos arquivos de dados da execução e reprodução do SciEvol apoiado pelo SGWfC Vistrails usando 8 núcleos de processamento.

A análise do uso e produção dos arquivos no SciEvol foi ampliada para ambientes que usam 16 e 32 núcleos para a execução do *workflow*. Nesse caso, não foi possível executar testes no Vistrails, pois o SGWfC não foi nativamente desenvolvido para a execução em ambientes que possuem a distribuição de nós sob uma rede de computadores. As Figuras 5.15 e 5.16 apresentam o número de arquivos manipulados pelo SciCumulus durante a execução do SciEvol usando 16 e 32 núcleos, respectivamente. É importante observar que o experimento original foi concebido após a execução de um conjunto de testes e reproduções nos cenários com 8, 16 e 32 núcleos. As reproduções foram executadas em cenários equivalentes, porem, nos provedores de nuvem. Em seguida, as reproduções foram comparadas as execuções, considerando a quantidade de núcleos equivalentes.

O quantitativo de elementos similares permite extrair a proporção da equivalência dos valores da descrição e *hash* em relação a quantidade de arquivos válidos.

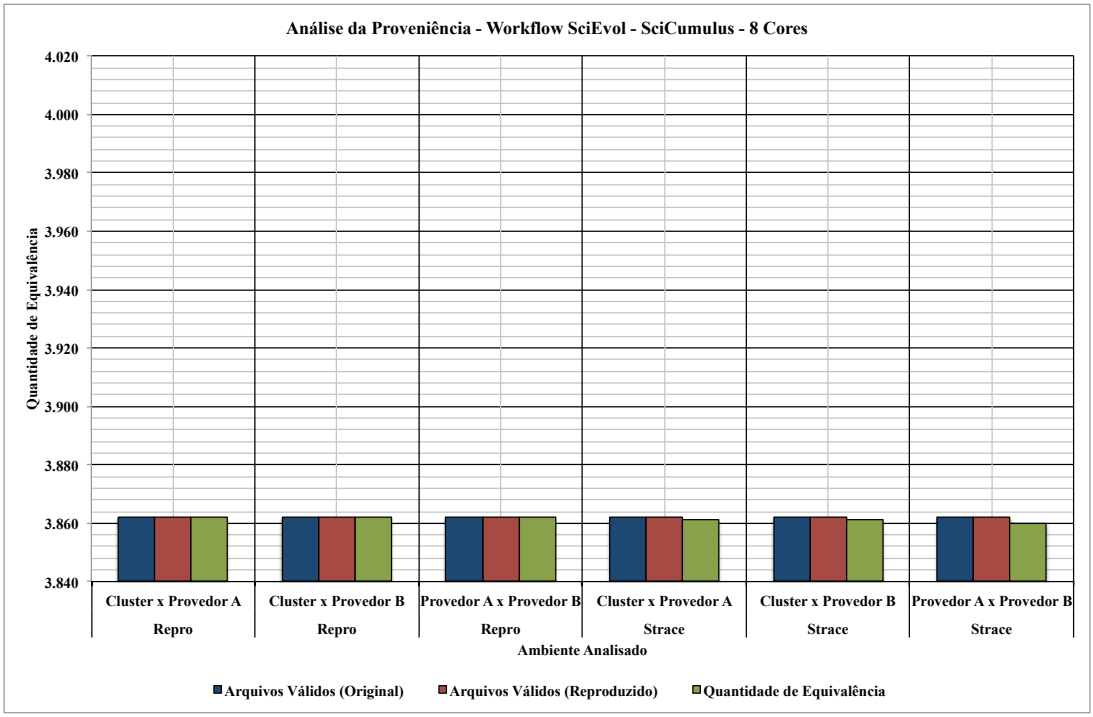


Figura 5.14: Equivalência dos arquivos de dados da execução e reprodução do SciEvol apoiado pelo SGWfC SciCumulus usando 8 núcleos de processamento.

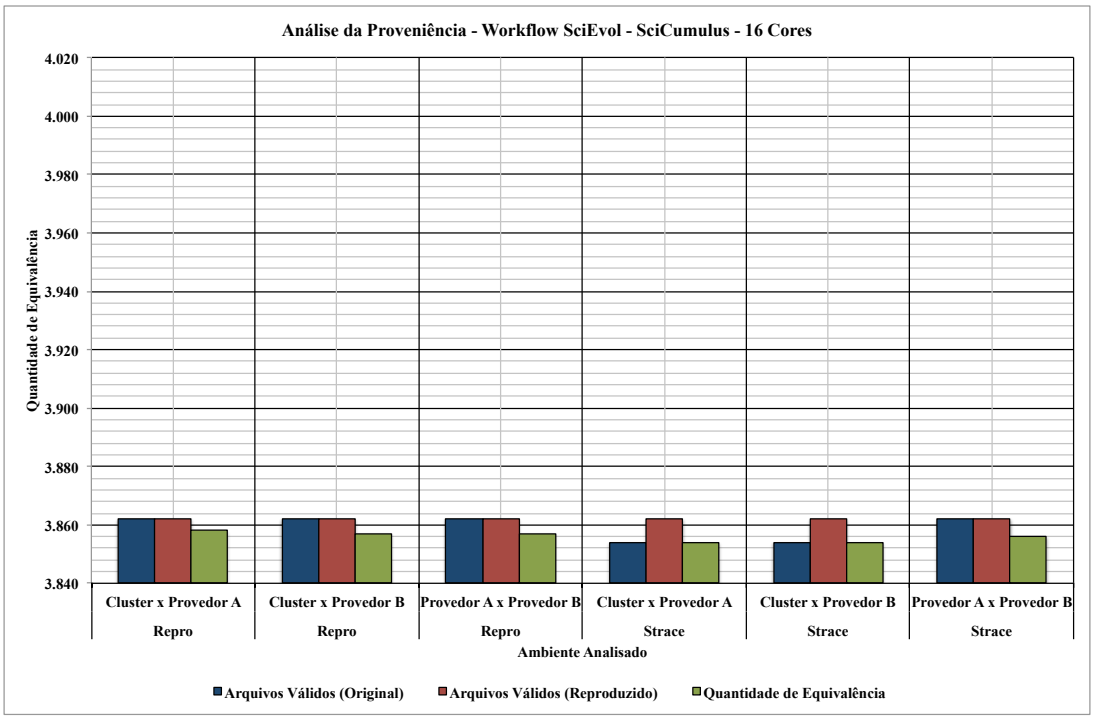


Figura 5.15: Equivalência dos arquivos de dados da execução e reprodução do SciEvol apoiado pelo SGWfC SciCumulus usando 16 núcleos de processamento.

O gráfico da Figura 5.17 mostra a taxa de reprodução extraída dos quantitativos de uso e geração de entidades arquivos obtidos pelos monitores *Strace* e *Repro*. Pode ser observado que em grande parte os experimento foram reproduzidos em valores

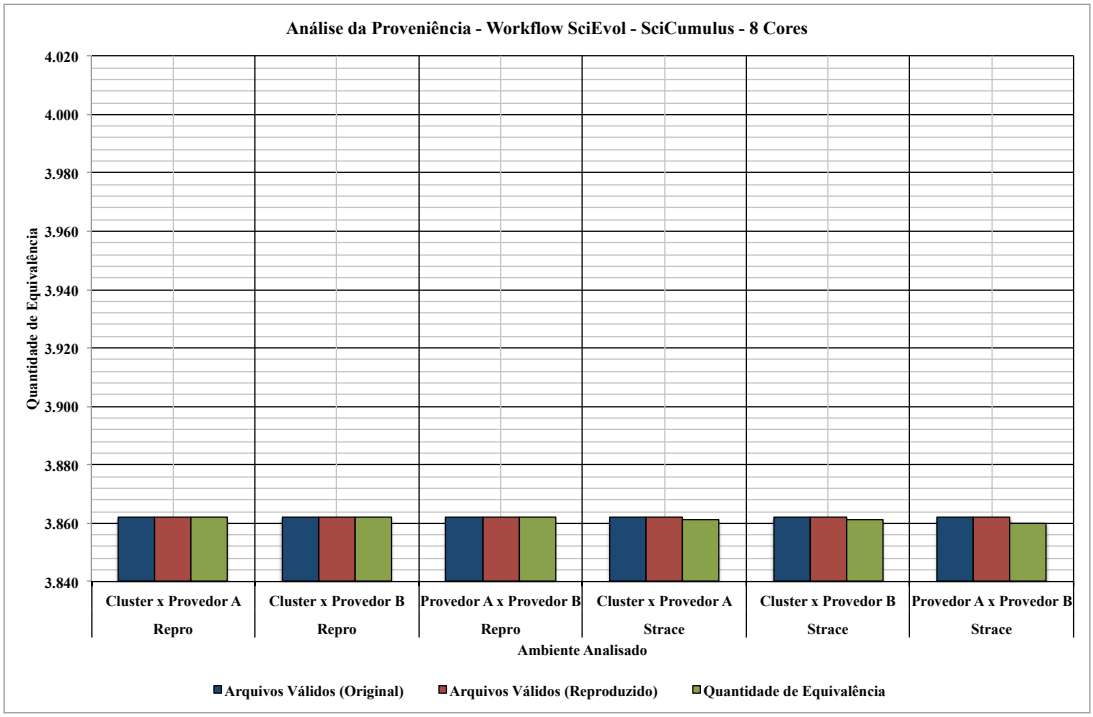


Figura 5.16: Equivalência dos arquivos de dados da execução e reprodução do SciEvol apoiado pelo SGWfC SciCumulus usando 32 núcleos de processamento.

superiores a 98% em todos os casos. Além disso, pode ser verificada a manutenção do padrão da taxa de reprodução das estratégias de monitoramento independente das alterações da quantidade de núcleos empregados na execução dos experimentos.

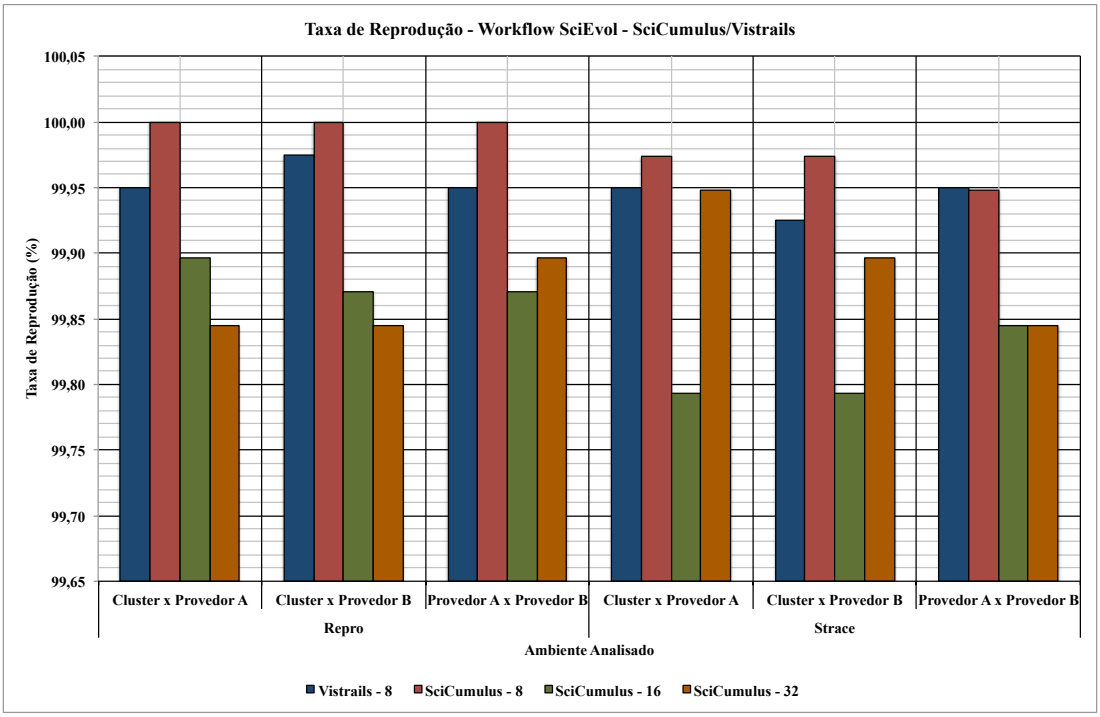


Figura 5.17: Equivalência dos arquivos na execução e reprodução do SciEvol.

Produção dos Elementos W3C Prov - *Workflow* Montage

As execuções do Montage produziram volumes de dados na escala de dezenas de gigabytes por execução. Grande parte da produção de arquivos originaram das atividades *Projection* e *FitPlane*. A Figura 5.18 mostra o grafo de proveniência gerado em uma execução do Montage. As arestas na cor azul representam os relacionamentos de uso (*used*) e as arestas em vermelho são as gerações (*wasGeneratedBy*). É possível verificar ainda arestas na cor verde. Elas representam o relacionamento de comunicação entre as atividades, onde a atividade predecessora foi comunicada pela antecessora sobre as entidades que ela gerou e que estão prontas para serem usadas.

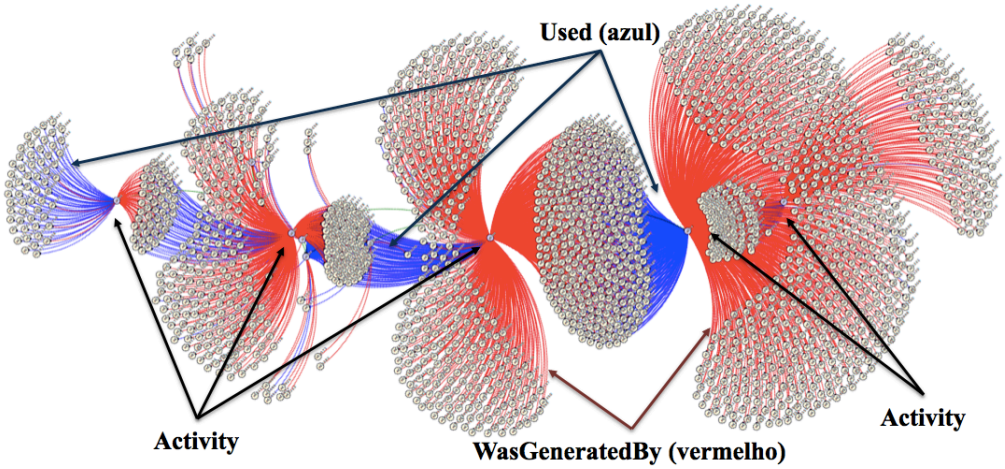


Figura 5.18: Grafo de proveniência produzido com a execução do *workflow* Montage apoiado pelo SGWfC SciCumulus.

O quantitativo de elementos W3C Prov gerados pelas execuções do Montage são apresentados na Tabela 5.5. Nesta tabela são mostradas as execuções monitoradas pelas estratégias *Strace* e *Repro* sob os 3 cenários de execução do Montage. Foram testados ambientes de computação com 8, 16 e 32 núcleos de processamento. É possível verificar que o quantitativo de elementos tem uma alteração de uma execução para a outra em todos os casos apresentados na tabela. Isso ocorre devido a característica da geração dos arquivos no *workflow*, que tem a sua criação orientada pelo conteúdo de arquivos previamente analisados.

A exemplo do SciEvol, o quantitativo de atividades e agentes no Montage, assim como os relacionamento de atribuição e delegação foram omitidos na tabela, isso se deve ao fato de que as execuções geram instâncias e valores constantes desses elementos nas execuções. Neste caso, também foram definidos 2 agentes, o usuário do sistema operacional e o próprio SGWfC, onde o primeiro agente delega a execução do *workflow* para o segundo.

Tabela 5.5: Dados da produção de elementos W3C Prov da execução (cluster) e reprodução (Provedores A e B) do Montage.

8 Cores						
Elementos	**Cluster**		**Provedor A**		**Provedor B**	
W3C Prov	**Strace**	**Repro**	**Strace**	**Repro**	**Strace**	**Repro**
Entities	20.931	20.892	20892	20.895	20.968	20.916
Used	18.533	18.492	18492	18.493	18.564	18.512
Generated	17.027	16.991	16991	16.993	17.054	17.011
Informed	12.990	12.958	12958	12.958	13.010	12.972
Attributed	17.027	16.991	16991	16.993	17.054	17.011
16 Cores						
Elementos	**Cluster**		**Provedor A**		**Provedor B**	
W3C Prov	**Strace**	**Repro**	**Strace**	**Repro**	**Strace**	**Repro**
Entities	20.912	20.995	20.884	20.929	20.934	20.890
Used	18.518	18.583	18.498	18.529	18.532	18.500
Generated	17.007	17.074	16.985	17.020	17.025	16.989
Informed	12.970	13.012	12.958	12.976	12.978	12.958
Attributed	17.007	17.074	16.985	17.020	17.025	16.989
32 Cores						
Elementos	**Cluster**		**Provedor A**		**Provedor B**	
W3C Prov	**Strace**	**Repro**	**Strace**	**Repro**	**Strace**	**Repro**
Entities	20.975	20.977	20.939	20.920	20.898	20.941
Used	18.583	18.593	18.553	18.538	18.520	18.561
Generated	17.056	17.058	17.027	17.011	16.993	17.030
Informed	13.000	13.010	12.980	12.970	12.958	12.988
Attributed	17.056	17.058	17.027	17.011	16.993	17.030

Análise da Equivalência da Proveniência - *Workflow* Montage

As Figuras 5.19, 5.20 e 5.21 apresentam a relação da geração e uso dos arquivos na execução do Montage gerenciado pelo SciCumulus. Essas informações refletem o percentual de equivalência do nome, *hash* e combinação dos dois atributos nos grafos de proveniência do *workflow*, obtidas pelos monitores implementados pelo ReproeScience. Esse gráfico foi construído com base nos valores apresentados na Tabela 5.6. A exemplo do SciEvol foram analisadas as execuções sob as duas estratégias de monitoramento. Ambas retornaram um conjunto de dados de monitoramento bastante similares, ou seja, os dados obtidos por uma estratégia confirma os dados coletados pela outra. A comparação dos ambientes foi feita de forma análoga, ou seja, *cluster* em relação aos provedores de nuvem, e um provedor de nuvem em relação ao outro.

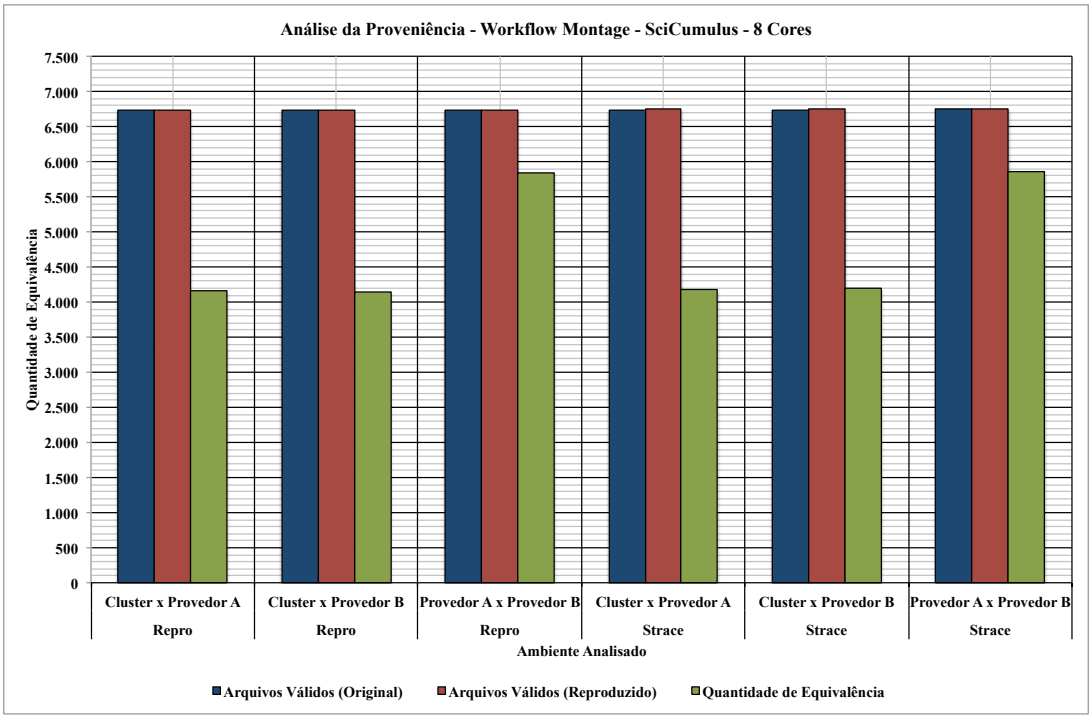


Figura 5.19: Equivalência dos arquivos de dados pela execução e reprodução do Montage apoiado pelo SGWfC SciCumulus.

Os gráficos mostram que as reproduções no Montage, originalmente executadas em um *cluster* em relação a nuvem, possuem uma menor taxa de equivalência se comparados com execuções em um provedor seguida da reprodução em outro. Em ambas as abordagens, *Strace* e *Repro*, as taxas de reprodução foram mais elevadas no segundo cenário do que no primeiro. Uma possível explicação pode estar relacionada com a precisão em que os recursos computacionais operam em cada ambiente. O ambiente computacional no *cluster* pode ter características específicas em relação a maior ou menor precisão das operações de ponto flutuante. Por esse motivo, os dados apresentados na Tabela, possuem maior acento na quantidade de vértices e

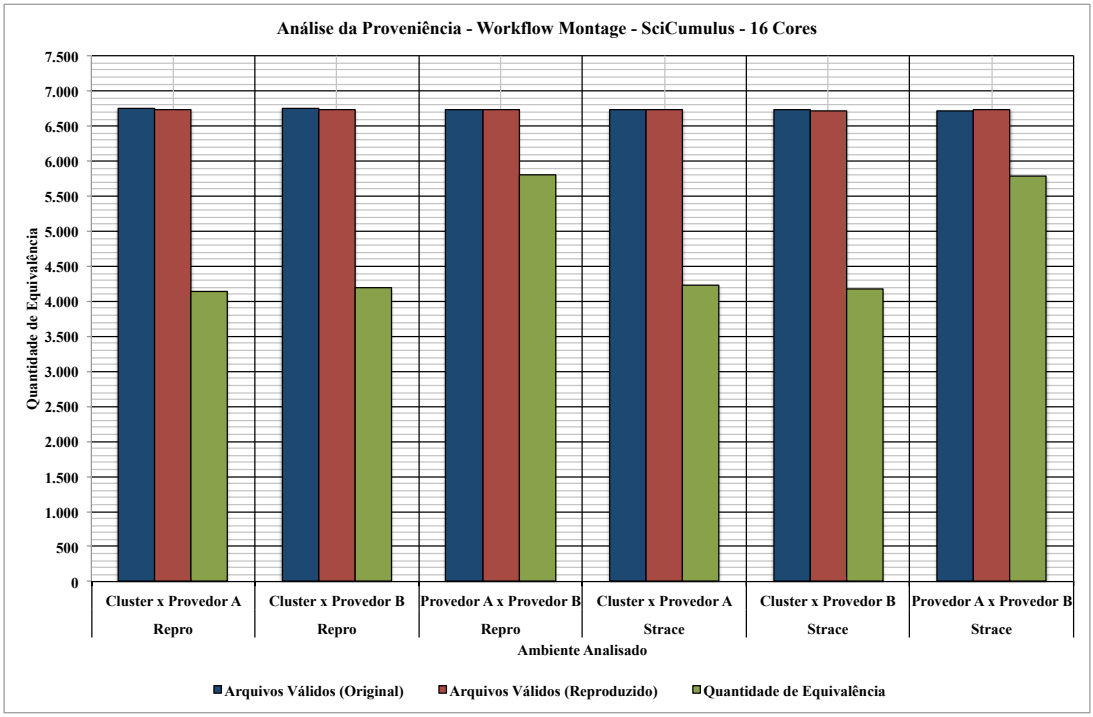


Figura 5.20: Equivalência dos arquivos de dados pela execução e reprodução do Montage apoiado pelo SGWfC SciCumulus.

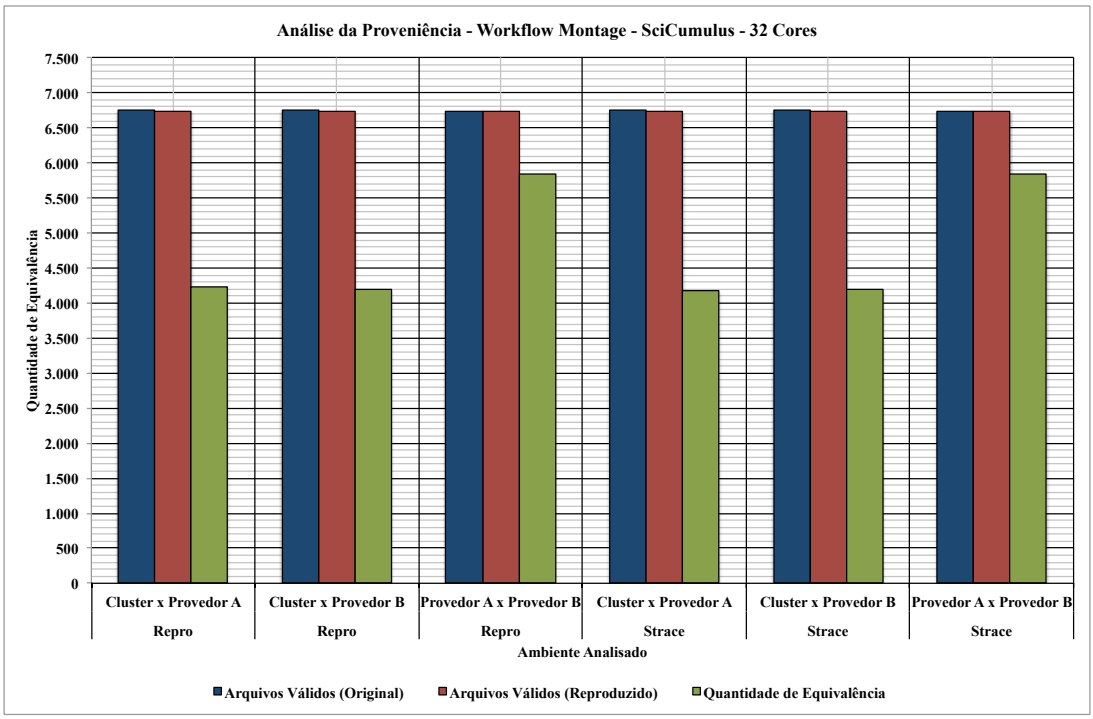


Figura 5.21: Equivalência dos arquivos de dados pela execução e reprodução do Montage apoiado pelo SGWfC SciCumulus.

arestas equivalentes, e também a quantidade de arquivos com o mesmo nome, vindo a valor do *hash* ter maior divergência.

Outro ponto a se analisar é o fato de que a geração de novos arquivos em uma

atividade corrente no Montage é orientada pelos dados contidos em arquivos usados em atividades anteriores, ou seja, a regra para a criação dos arquivos em uma atividade é orientada pelo conteúdo dos arquivos gerados na atividade anterior. Como as operações do *workflow* geram valores aproximados, as características computacionais de um ambiente pode interferir nessa criação de arquivos, o processamento em um ambiente pode determinar a criação de um dado arquivo enquanto no outro ambiente essa criação não foi determinada. Ambientes com características mais similares podem gerar uma maior quantidade de arquivos equivalentes e, consequentemente, taxas de reprodução mais elevadas.

Ao observar o gráfico da Figura 5.22 é possível verificar que ao comparar as reproduções obtidas nos provedores de nuvem em relação a execução no *cluster* as equivalências ficaram em aproximadamente 60%. Ao observar as taxas de reprodução de um provedor de nuvem em relação ao outro pode-se constatar que essa taxa fica mais elevada, em torno de 80%. É possível observar ainda que essa taxa de reprodução se mantém independente da quantidade de núcleos alocados. O Montage gera um grafo de equivalência com característica de subgrafos, ou seja, o número de entidades arquivos lidos e gerados são diferentes de uma execução para a outra e as informações de nome de arquivos e *hash* diferenciam de 60 a 80%, mesmo usando ambientes similares e os mesmos conjunto de dados.

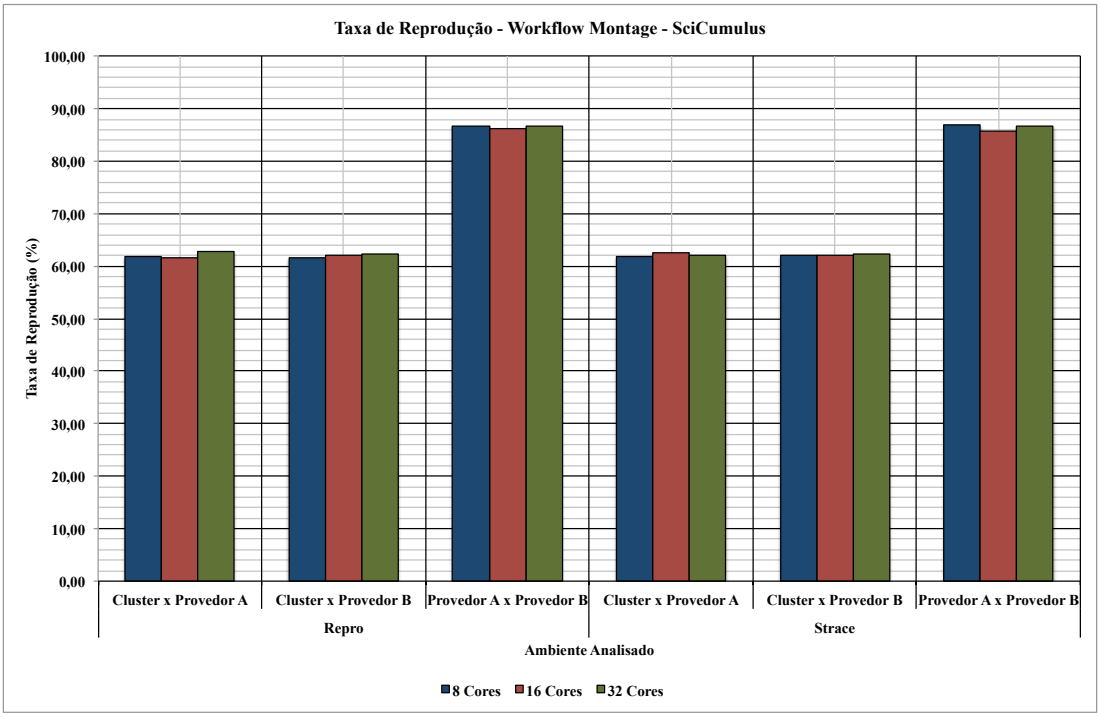


Figura 5.22: Equivalência dos arquivos na execução e reprodução do Montage.

Tabela 5.6: Taxa de reprodução de arquivos do *Workflow* Montagem com o SciCumulus.

Estratégia	**Ambiente Original x Reproduzido**	**Arquivos Válidos (Original)**	**Arquivos Válidos (Reproduzido)**	**Nome Equivalente**	**Hash Equivalente**	**Nome/Hash Equivalentes**	**Quantidade de Equivalência**	**Taxa de Reprodução (%)**
Repro	**Cluster x Provedor A**	6.728	6.736	3.520	0	644	4.164	61,82
	Cluster x Provedor B	6.728	6.729	3.501	0	643	4.144	61,58
	Provedor A x Provedor B	6.729	6.736	1.261	1.955	2.628	5.844	86,76
Strace	**Cluster x Provedor A**	6.735	6.750	3.529	0	644	4.173	61,82
	Cluster x Provedor B	6.735	6.755	3.548	0	644	4.192	62,06
	Provedor A x Provedor B	6.755	6.750	1.246	1.978	2.639	5.863	86,86
16 Cores - SciCumulus								
Estratégia	**Ambiente Original x Reproduzido**	**Arquivos Válidos (Original)**	**Arquivos Válidos (Reproduzido)**	**Nome Equivalente**	**Hash Equivalente**	**Nome/Hash Equivalentes**	**Quantidade de Equivalência**	**Taxa de Reprodução (%)**
Repro	**Cluster x Provedor A**	6.756	6.727	3.498	0	647	4.145	61,62
	Cluster x Provedor B	6.756	6.738	3.541	0	645	4.186	62,13
	Provedor A x Provedor B	6.756	6.725	1.271	1.941	2.591	5.804	86,28
Strace	**Cluster x Provedor A**	6.733	6.737	3.577	0	646	4.223	62,68
	Cluster x Provedor B	6.733	6.725	3.525	0	644	4.169	61,99
	Provedor A x Provedor B	6.725	6.737	1.311	1.980	2.581	5.782	99,95
32 Cores - SciCumulus								
Estratégia	**Ambiente Original x Reproduzido**	**Arquivos Válidos (Original)**	**Arquivos Válidos (Reproduzido)**	**Nome Equivalente**	**Hash Equivalente**	**Nome/Hash Equivalentes**	**Quantidade de Equivalência**	**Taxa de Reprodução (%)**
Repro	**Cluster x Provedor A**	6.752	6.740	3.589	0	647	4.236	62,85
	Cluster x Provedor B	6.752	6.735	3.546	0	643	4.189	62,20
	Provedor A x Provedor B	6.735	6.740	1.287	1.927	2.627	5.841	86,66
Strace	**Cluster x Provedor A**	6.750	6.729	3.536	0	643	4.179	62,10
	Cluster x Provedor B	6.750	6.739	3.548	0	645	4.193	62,22
	Provedor A x Provedor B	6.739	6.729	1.246	1.952	2.640	5.838	86,76

Considerações sobre a Equivalência da Reprodução

É importante ressaltar que a abordagem informa o percentual de reprodução no conteúdo dos dados de entrada, intermediários e resultados finais. Além disso, informa o percentual de reprodução do grafo de proveniência, através do isomorfismo de grafos e sub grafos. Em termos de isomorfismo ocorreu uma equivalência superior a 95%, tanto no *workflow* SciEvol quanto no Montage. Esses dados podem verificados, respectivamente, nas Tabelas 5.3 (SciEvol) e 5.5 (Montage). Esse isomorfismo é reforçado pelo fato das arestas dos grafos de proveniência terem informações semânticas relacionadas sobre tipos das ligações.

5.4 Análise do Desempenho

O principal objetivo da adoção dos mecanismos de apoio a ambientes HPC para execução de um experimento, é a obtenção de um aumento no desempenho do processamento de uma análise. Quando um cientista produz uma pesquisa que precisa do apoio deste tipo de ambiente é indispensável prover mecanismos para que outros usuários possam tirar as mesmas vantagens deste tipo de ambiente. Essa foi uma das motivações da proposta ReproeScience, permitir que terceiros tenham acesso a infraestruturas que permitam a reprodução de experimentos que utilizam recursos de HPC no processamento das suas análises. Diante disso, é necessário verificar se a solução de reprodução mantém níveis de desempenho satisfatórios em relação a execução do experimento original, ou seja, é preciso manter níveis de desempenho que garantam o processamento da reprodução em um tempo hábil, a exemplo da execução original do experimento.

Os experimentos apoiados pelo ReproeScience tiveram o tempo de execução obtidos e armazenados para fins de verificação do comportamento de execução dos *workflows* científicos nos provedores de nuvem em relação a infraestrutura do *cluster* local. Conforme mostrado na Seção 5.2, foram selecionados recursos de computação com características mais próximas possíveis, para se obter uma análise de desempenho em condições equivalentes. Os desempenho foi avaliado com base no tempo de execução, a partir de uma média extraída após quatro rodadas de ensaios para cada cenário 8, 16 e 32 núcleos. Os experimentos foram realizados em infraestruturas com recursos de computação de 8, 16 e 32 núcleos, distribuídos, respectivamente, em 1, 2 e 4 nós de processamento. As próximas seções discutem os comportamentos de execução dos *workflows* dos estudos de caso.

Foram executadas 5 rodadas de testes para cada cenário apresentado. A melhor e a pior execução de cada *workflow* foram eliminadas e, a partir disso, foram consideradas a média das 3 execuções que sobraram. Devido ao alto custo computacional,

experimentos levaram um tempo considerável para concluir as execuções. Diante disto, os resultados das execuções são apresentados em horas, minutos e segundos. O objetivo dessas análises limita-se a mostrar que a reprodução em um ambiente de nuvem também é viável no ponto de vista de desempenho e não para comparar qual ambiente possui o melhor desempenho. Como a nuvem oferece recursos pré configurados, não é possível obter uma infraestrutura completamente similar ao *cluster* usado nos experimentos, ou até mesmo entre os provedores, inviabilizando uma análise de desempenho com o objetivo de definir o melhor ambiente de execução.

5.4.1 Desempenho do *Workflow* SciEvol

Os experimentos no SciEvol foram executados nos ambientes de SGWfC SciCumulus e Vistrails. A Figura 5.23 mostra o comportamento da execução do *workflow* no Vistrails. Nesse caso é possível observar o pico de processamento nas atividades *ReadSeq* e *Codeml*, essa última, com características de alto consumo de recursos de processador. Nesse caso, pode ser observado que o provedor B obteve os piores resultados nas atividades com maior consumo de recursos. Entretanto, o provedor A, apresentou bons tempos de execução, muitas vezes mais eficientes que o *cluster* local. A avaliação do Vistrails limitou-se ao cenário de 1 nós (8 núcleos).

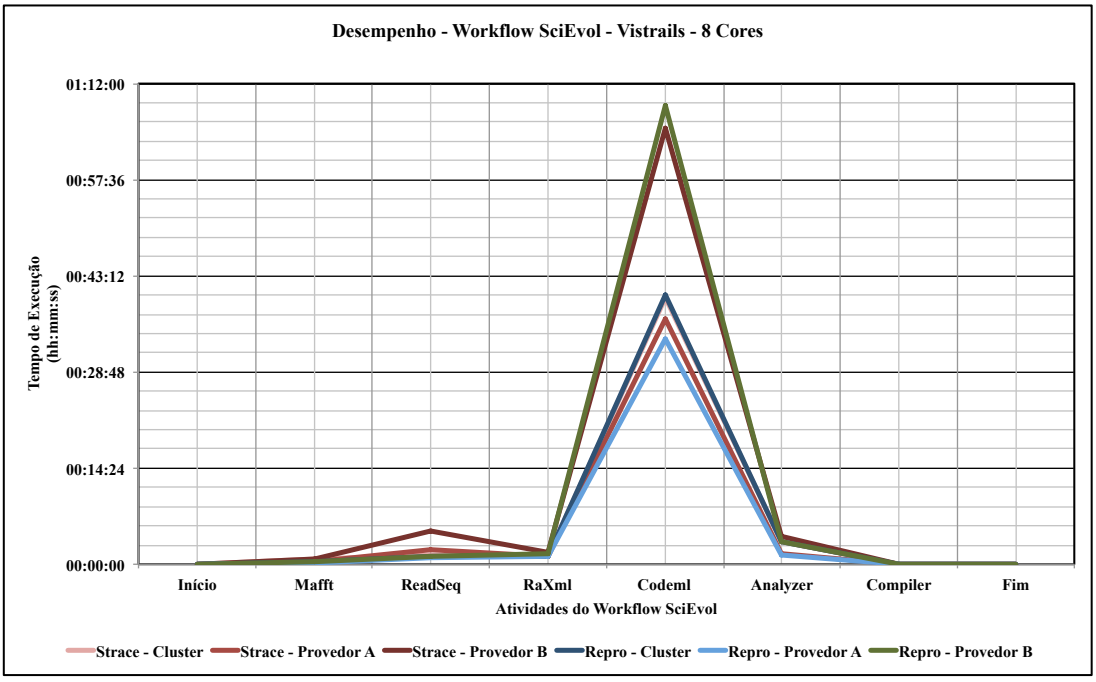


Figura 5.23: Desempenho da execução e reprodução do SciEvol no Vistrails com 8 núcleos.

Os testes realizados no SciCumulus apresentaram os picos nos tempos de execução nas atividades relacionadas ao *Codeml*. Conforme esperado, a medida que o número de núcleos aumentou, houve uma queda no tempo de execução dos

workflows. O ambiente com 8 núcleos apresentou picos de processamento entre 18 e 36 minutos, enquanto a execução com 16 núcleos ficou entre 8 e 13 minutos, reduzindo significativamente o tempo de execução. Comparado com as execuções com 32 núcleos, esse tempo teve ainda mais queda na atividade com maior pico, ficando entre 7 e 12 minutos.

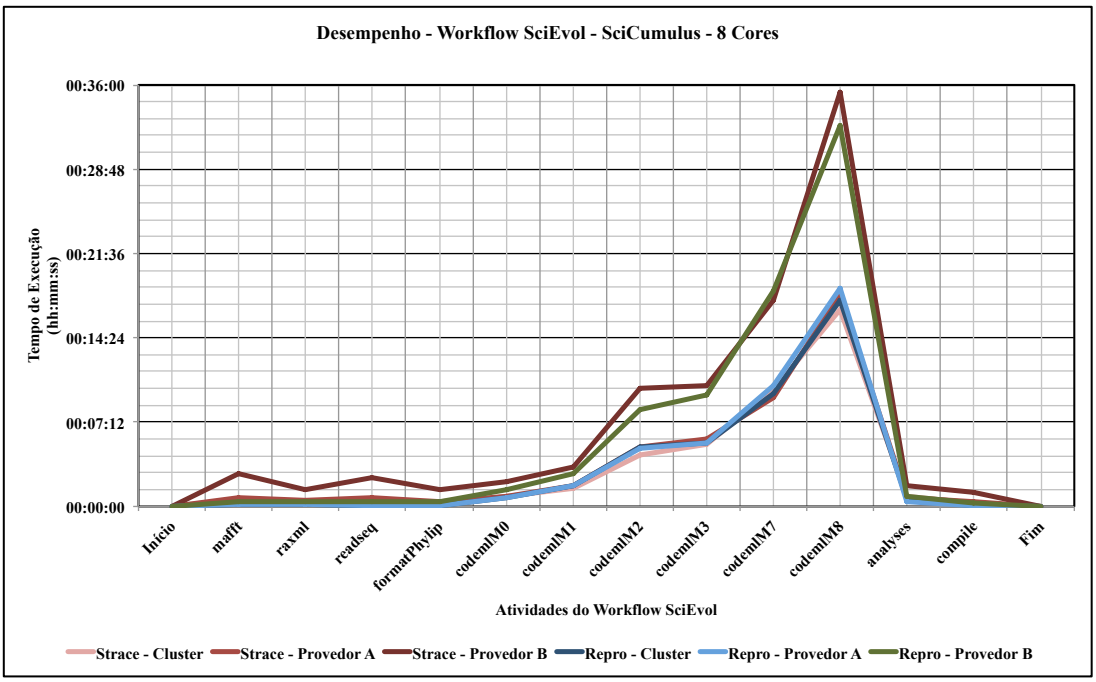


Figura 5.24: Desempenho da execução e reprodução do SciEvol no SciCumulus com 8 núcleos.

O provedor de nuvem B continuou a registrar um desempenho inferior, no entanto, manteve o padrão de redução no tempo de execução do *workflow* a medida que os recursos computacionais aumentaram. O SciEvol possui a característica de ser intensivo em processamento, demandando uma elevada quantidade de recursos de processamento. Observando os resultados pode-se verificar que o provedor A manteve os tempos de execução na mesma faixa que os obtidos com o *cluster*, portanto, em termos de desempenho, os experimentos mostraram que as reproduções de experimentos computacionais na nuvem atende as expectativas de ganho computacional no processamento de *workflows* que demandam por recursos de HPC.

A Figura 5.27 apresenta o comportamento do desempenho das estratégias de monitoramento implementadas pelo ReproeScience. As barras em azul mostram o tempo de execução da estratégia *Repro*, enquanto as barras em vermelho mostram o desempenho da estratégia *Strace*. As duas estratégias foram avaliadas em relação a execução sob os dois SGWfC usados para o *workflow* SciEvol.

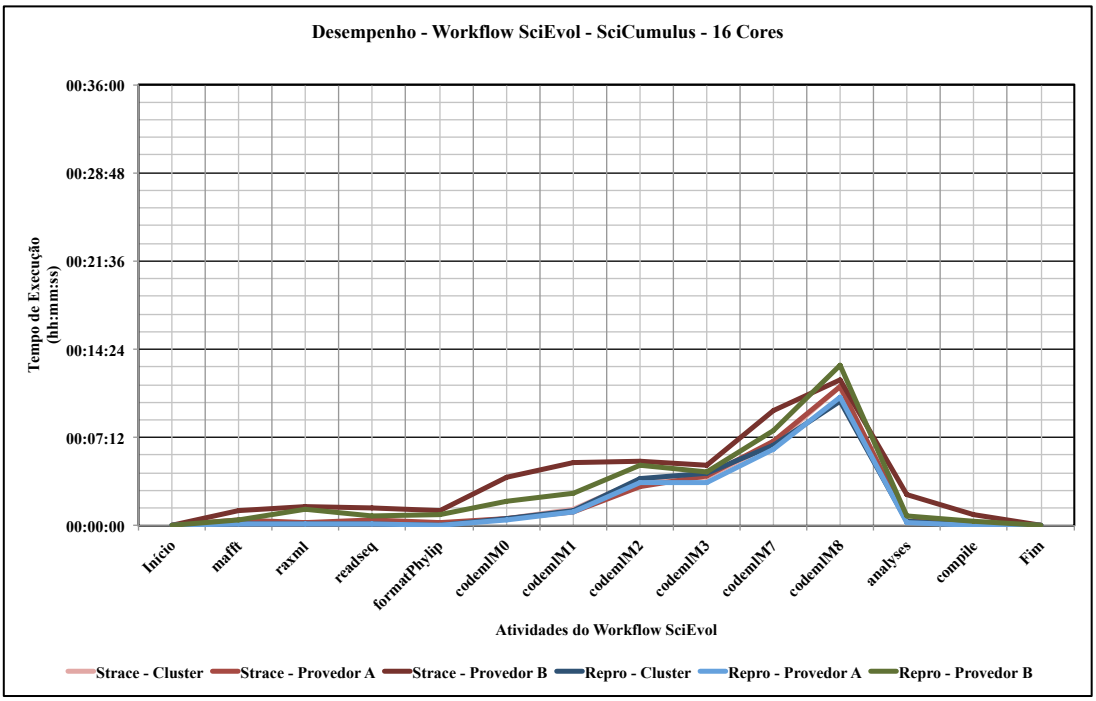


Figura 5.25: Desempenho da execução e reprodução do SciEvol no SciCumulus com 16 núcleos.

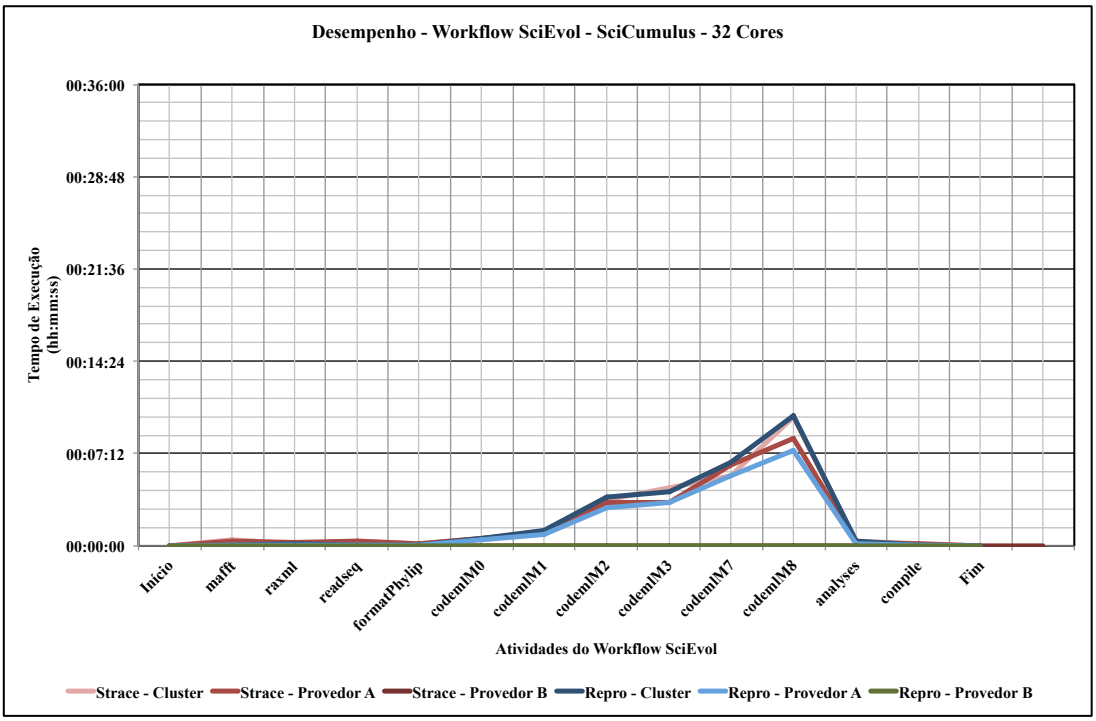


Figura 5.26: Desempenho da execução e reprodução do SciEvol no SciCumulus com 32 núcleos.

5.4.2 Desempenho do *Workflow* Montage

Os experimentos realizados no Montage receberam o suporte do SGWfC SciCumulus para a execução das 9 atividades em paralelo. A principal característica do Montage

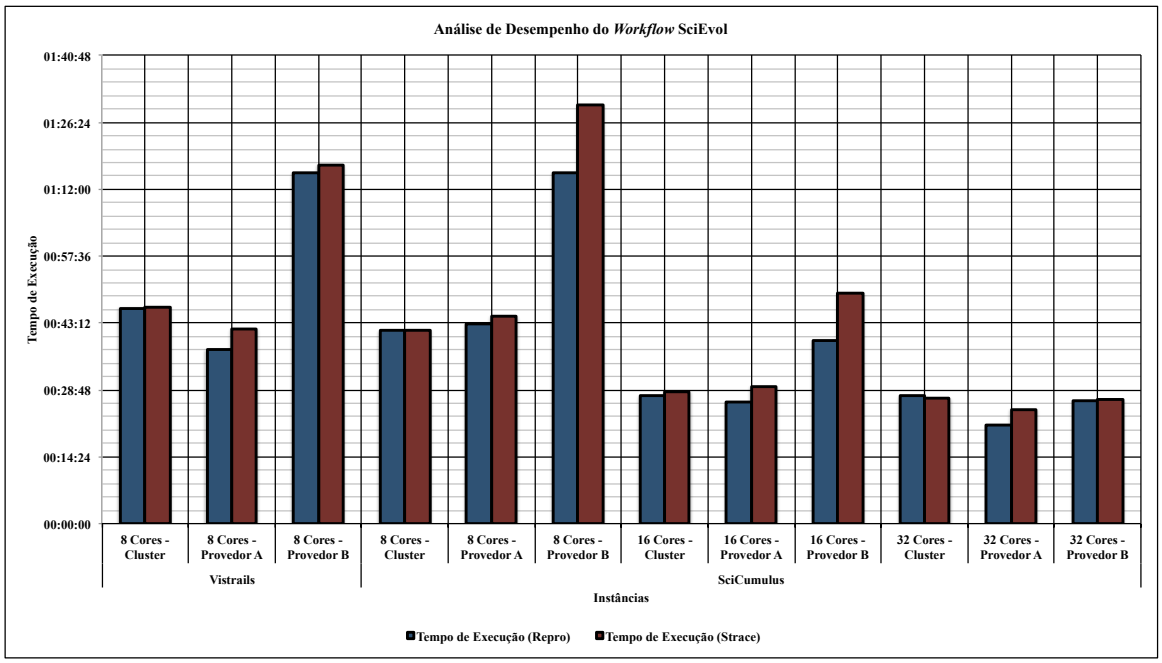


Figura 5.27: Tempo total de execução do SciEvol no *cluster* e provedores de nuvem com recursos de computação do Vistrails 8 núcleos e SciCumulus 8, 16 e 32 núcleos.

é a intensividade de dados, ou seja, as atividades consomem e produzem um grande número de informações de arquivos de dados. Foi observado nos experimentos que o SciEvol precisa de otimização em recursos de processamento, já o Montage precisa de investimento em recursos de otimização em disco e redes de computadores, esse último, quando o sistema de armazenamento é remoto e ocorre altas taxas de transferência de dados na rede até o equipamento de armazenamento.

Analisando os gráficos de desempenho é possível observar que a atividade *Select-Projections* demanda o maior tempo no experimentos, salvo o caso com 8 núcleos em que a atividade *CalculareDifferences* apresentou um pico de execução. É importante observar que nos três cenários de execução, 8, 16 e 32 núcleos, os provedores de nuvem obtiveram um bom desempenho quando comparados com a execução no *cluster*. A exemplo do caso do SciEvol, o provedor B obteve um resultado menos significativo, no entanto, atendeu a expectativa de diminuir o tempo de processamento a medida que ocorre o aumento do número de recursos computacionais.

Foi observado nos experimentos que a estratégia que implementa o *Strace* possui um desempenho um pouco inferior em relação a estratégia *Repro*, principalmente no caso das execuções do Montage. Essa a diferença de tempo pode ter sido maior devido ao grande número de programas e arquivos executados pelo Montage. O *Strace* obtém informações mais detalhadas sobre os processos e o sistema de arquivos por meio de chamadas de sistema e interrupções, ao contrário da estratégia *Repro*. Além disso, o *Strace* intercepta todas as chamadas atuando como uma espécie de depurador (*debug*). Desta forma, cada chamada de arquivos e processos deve ser

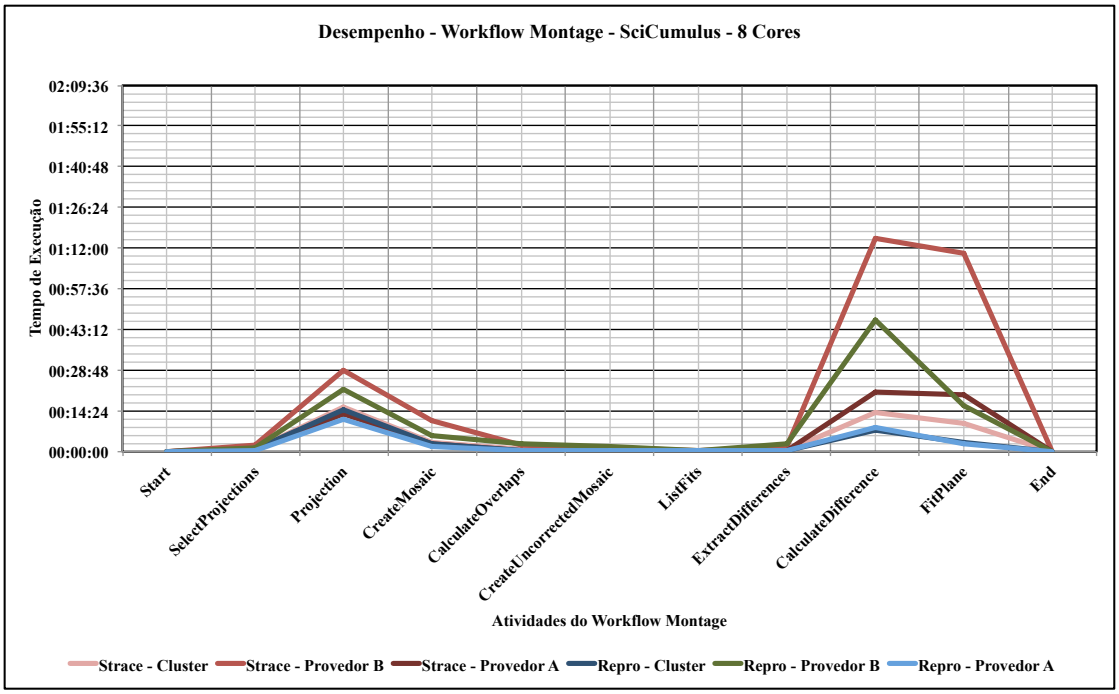


Figura 5.28: Desempenho da execução e reprodução do Montage com 8 núcleos.

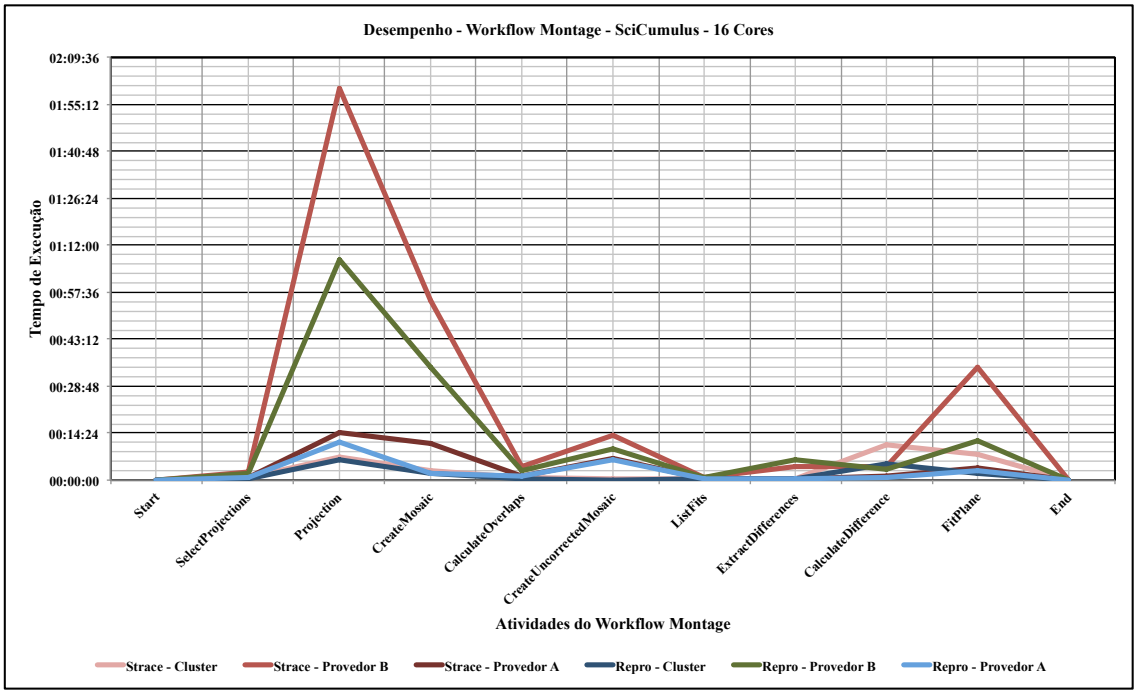


Figura 5.29: Desempenho da execução e reprodução do Montage com 16 núcleos.

tratada pelo *Strace* antes de ser realmente executada. Como a estratégia *Repro* consome as informações de uma fila de registros de acessos a arquivos e processos sem interferir na execução, o monitoramento não prejudica tanto o desempenho.

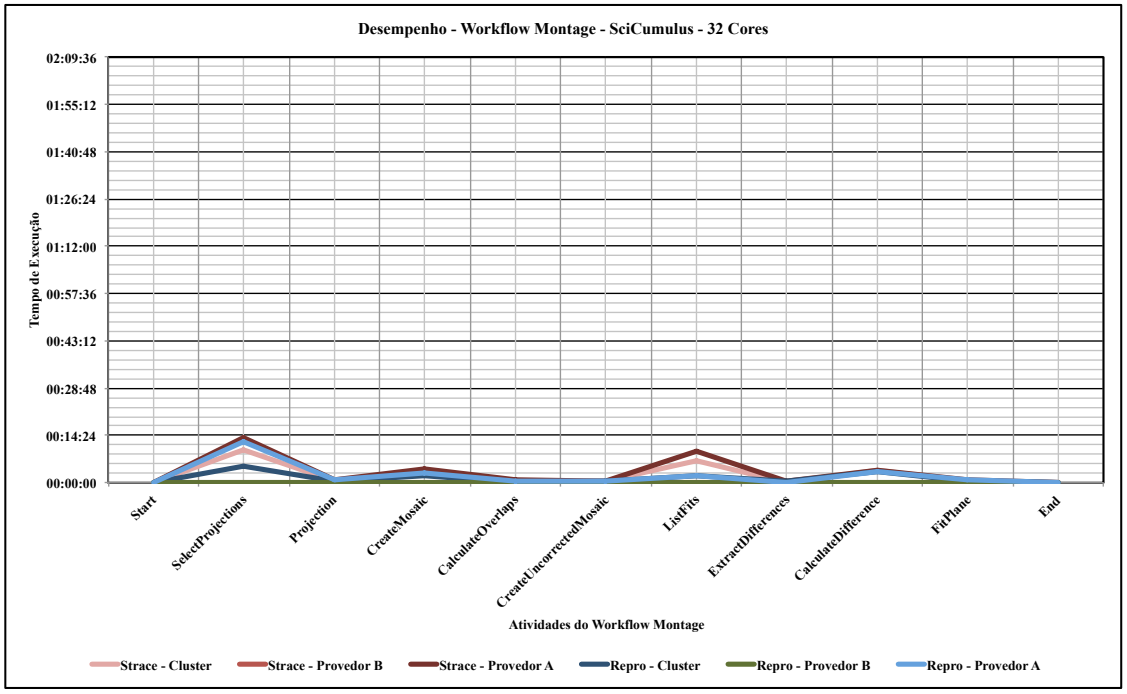


Figura 5.30: Desempenho da execução e reprodução do Montage com 32 núcleos.

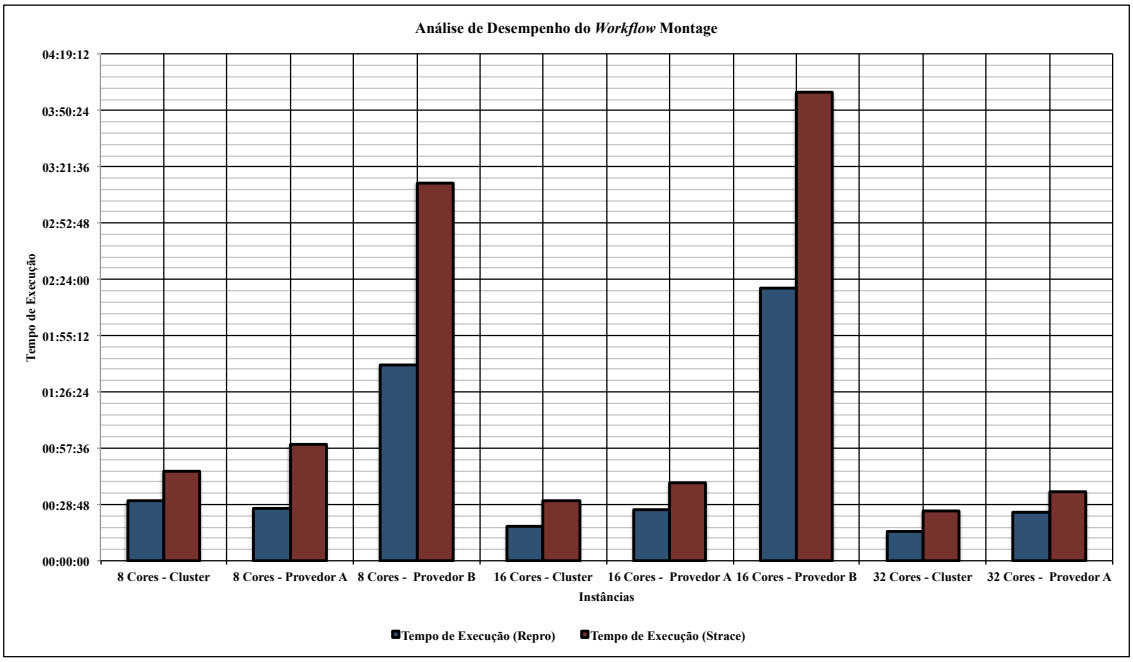


Figura 5.31: Tempo total de execução do Montage no *cluster* e provedores de nuvem com recursos de computação de 8, 16 e 32 núcleos.

5.4.3 Considerações sobre o Desempenho nos Experimentos

Nos primeiros experimentos foi observado que o tempo de execução em um dos provedores de nuvem selecionados foi excessivamente alto. Foi constatado que a máquina em execução passava mais tempo trabalhando em tarefas relacionadas a entrada e saída de dados (I/O) ao invés de efetivamente usar os processadores na

sua máxima capacidade. Como estava sendo empregado um computador como servidor de arquivo, foi possível monitorar a velocidade do fluxo de dados na rede e, consequentemente, o fluxo para leitura e gravação de arquivos. O monitoramento do fluxo de dados foi realizado por meio do aplicativo *speedometer* gerando os gráficos de atividades a exemplo dos apresentados na Figura 5.32.

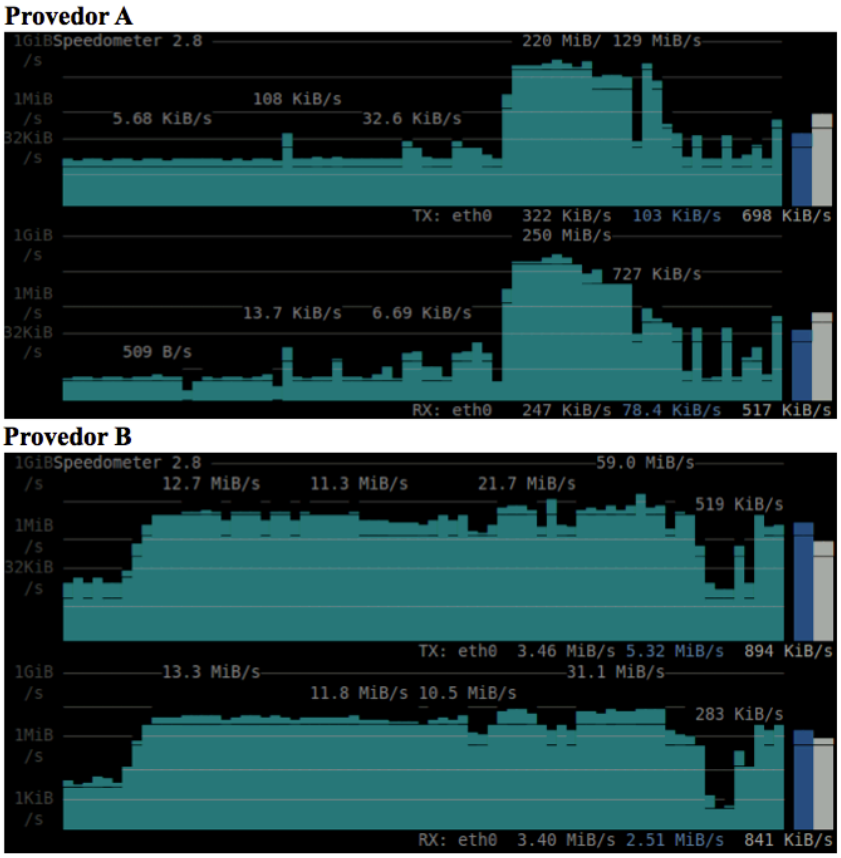


Figura 5.32: Exemplo de fluxo de I/O da execução de atividades do SciEvol.

Estes gráficos apresentam os resultados do monitoramento do fluxo de dados em uma mesma atividade nos provedores A e B. O eixo x dos gráficos apresenta o tempo demandado, no eixo y está a velocidade do fluxo de informação e as siglas TX e RX representam, respectivamente, a taxa de dados transmitidos e recebidos. É possível observar que a vazão de disco do provedor B (parte inferior) é constante e alta em relação a capacidade total do recurso, limitando TX em $59,0 Mbytes/s$, RX em $31,1 Mbytes/s$. Já no provedor A a taxa de transmissão é mais alta se comparado ao primeiro provedor, obtendo um TX em $220 Mbytes/s$ e um RX em $250 Mbytes$. Portanto, o gargalo da execução no provedor B são os recursos de I/O. Analisando os gráficos fica claro o quanto é primordial escolher recursos computacionais na nuvem com maiores taxa e velocidade de acesso ao disco, pois trata-se de um recurso que impacta diretamente no tempo total de execução do *workflow*.

Essa observação fica clara quando é analisada a execução dos experimentos com o

objetivo de aumentar os recursos para a redução do tempo de execução. A princípio, uma reprodução utiliza a mesma quantidade e características de recursos usados no experimento original. No entanto, um cientista pode ter o interesse de aumentar o número de recursos com o objetivo de diminuir o tempo de execução, uma vez que ele tem acesso a uma grande variedade de recursos dos provedores de nuvem. Por exemplo, no caso do Montage, que é um *workflow* intensivo de dados, a redução do tempo de execução não é proporcional ao aumento do número de recursos de processamento. A redução do tempo de execução é dependente da otimização dos recursos relacionados a disco.

Por exemplo, foi observado que um *workflow* que executou as suas atividades com 8 núcleos em 30 minutos obteve um ganho de 33% no tempo de execução ao executar o mesmo *workflow* com 32 núcleos, nesse caso, em 20 minutos. Levando em consideração que cada nó usado nos experimentos tinha 8 núcleos, será necessário multiplicar o valor dos recursos usados no primeiro cenário por 4 para alcançar os 32 núcleos do segundo cenário do experimento. Porém, a velocidade de acesso ao disco não cresce na mesma proporção que o número de núcleos adicionados ao experimento, portanto, pode ser necessário fazer uma opção por manter o número de núcleos baixo executando por um maior tempo ao invés de alocar mais recursos de processamento para melhorar o desempenho, já que se trata de uma reprodução.

É importante destacar que o provedor de nuvem oferece uma diversidade de recursos de computação. O cientista que vai reproduzir o experimento pode fazer a implantação com recursos com maior desempenho, porém, deve analisar os custos que serão despendidos com essa decisão. Portanto, a escolha do tipo de instância fica a cargo do experimentador. A abordagem sugere as instâncias baseada nos dados de proveniência sobre os perfis de computadores usados na execução do experimento original, para a produção dos resultados, porém, não é uma regra usar especificamente os recursos apontados.

A adoção do ReproeScience causa uma sobrecarga na execução do *workflow*, conforme mostrado na Figura 5.33. O percentual dessa sobrecarga também é apresentado na Tabela 5.7. As análises foram aplicadas aos cenários de 16 e 32 cores no ambiente de produção dos dados no *cluster*. São apresentados os comportamentos da execução tanto com a abordagem de monitoramento Repro quanto com a Strace. As barras em amarelo mostram o tempo de resposta das execuções apoiadas pelo ReproeScience, enquanto as barras em azul mostram as execuções sem esse apoio.

Diante desses resultados, é necessário fazer uma análise para verificar os impactos da sobrecarga causada pela adoção da solução de reprodução. Principalmente em que momento ela deve ser adotada, em todas as execuções ou somente no momento em que a execução de produção dos resultados for efetivada.

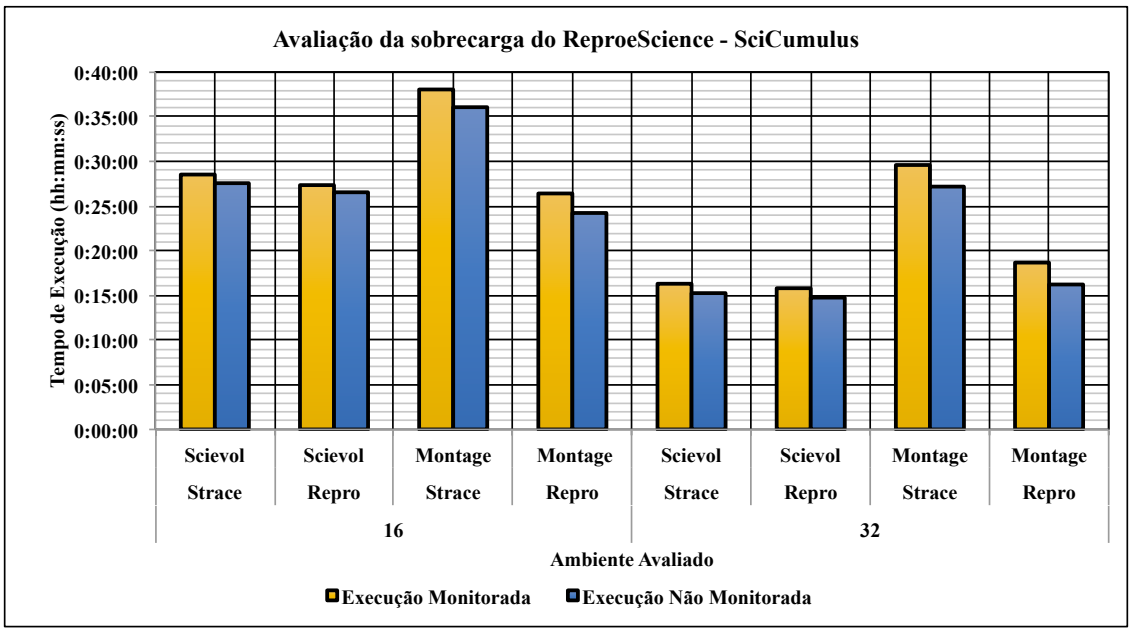


Figura 5.33: Sobrecarga da adoção do ReproeScience aos experimentos.

Tabela 5.7: Porcentagem de carga adicionada a execução dos *workflows* com a adoção do ReproeScience.

Cores	Monitor	Workflow	Execução Monitorada	Execução Não Monitorada	Diferença Tempo	Sobrecarga (%)
16	Strace	Scievol	00:28:32	00:27:33	00:00:58	3,51
	Repro	Scievol	00:27:20	00:26:32	00:00:48	3,02
	Strace	Montage	00:38:02	00:36:03	00:01:59	5,50
	Repro	Montage	00:26:24	00:24:13	00:02:11	9,02
32	Strace	Scievol	00:16:18	00:15:15	00:01:03	6,89
	Repro	Scievol	00:15:48	00:14:45	00:01:03	7,12
	Strace	Montage	00:29:35	00:27:10	00:02:25	8,9
	Repro	Montage	00:18:40	00:16:13	00:02:27	15,11

5.5 Comparação do ReproeScience com o Estado da Arte

O ReproeScience atende a grande parte das características das demais abordagens apresentadas nas Tabelas 3.1 e 3.2. No entanto, a abordagem possui diversas vantagen principalmente no que tange a portabilidade e acesso a infraestrutura de reprodução. Em termos de metodologia de reprodução, a adotada pelo ReproeScience é nativamente a reprodução, permitindo que as demais, auditável, confirmável, revisável e repetível, sejam instantaneamente agregadas. A abordagem proposta é orientada a dados e engloba todas as categorias de dados classificados no mapa conceitual.

Em termos de atores, a abordagem distingue apenas dois, o autor do experimento e os leitores. Os custos de publicação de uma pesquisa no ReproeScience incidem para o autor do experimento. Os revisores e editoras devem ter acesso as credenciais do autor para efetuar a reprodução e desta forma avaliar a publicação. Essa definição

repassa os custo da avaliação da reprodução para o autor da pesquisa. Já no caso dos leitores, os custos de reprodução da pesquisa ficam sob sua responsabilidade. A apresentação é feita por meio de um artigo PDF inserido em um texto escrito em Latex. Este documento é acompanhado de um XML que descreve os meios para acessar os objetos de pesquisa, e quando desejado, implanta a infraestrutura e reproduz o experimento. Conforme mostrado neste capítulo, a avaliação é orientada a validação dos metadados e dados de proveniência, e ainda dá suporte para a verificação analítica dos resultados produzidos.

O fator de licenciamento no ReproeScience possui um controle explicito que restrinje o acesso aos objetos de pesquisa baseado nos tipos de licença. É levado em conta que ao publicar um dado experimento, o autor tem o conhecimento que todos os elementos ficarão públicos a outros usuários. Existe a possibilidade de informar que tipo de licença está relacionada ao objeto, porém, o controle explícito não foi implementado. A característica sobre o *hardware* diferencia a abordagem das demais do estado da arte. O componente foi construído para contornar os desafios relacionados com ambientes de computação HPC e com grande volume de dados. Apesar de algumas outras abordagens atenderem aos requisitos de computação e armazenamento, estas especificidades não são tratadas por elas e, consequentemente, limitam a reprodução de uma pesquisa com tais características.

Quando uma pesquisa com o suporte do ReproeScience é implantada na nuvem, ela fica hospedada em uma VM gerenciada por um servidor de aplicação. Este servidor permite o acesso aos objetos de pesquisa para que as informações armazenadas nos dados e metadados apresentados possam ser apresentadas ao usuário. Além disso, o usuário pode obter o objeto de pesquisa por meio de *download* e incluir comentários sobre ele, tanto por parte dos autores quanto pelos demais usuários. Essa característica atende o fator de anotação relacionado a documentação da pesquisa. Todos os parâmetros e variáveis de ambiente, assim como todas as classes de *software*, são encapsulados em VMs na nuvem. Portanto, a abordagem atende a todas as características dos fatores configuração e *software*.

5.6 Limitações

Existem alguns fatores que podem limitar a execução e obtenção da reprodução dos experimentos com a abordagem. Alguns deles estão relacionados com a precisão dos resultados gerados na reprodução em relação a execução original do experimento. A reprodução pode não alcançar exatamente os mesmos resultados produzidos no experimento original. Diante disto, é necessário avaliar a precisão para verificar se os resultados estão dentro dos parâmetros aceitáveis pelos métodos adotados pelo cientista. Por exemplo, a ISO ISO/DTS 21748/2002 [45] determina que uma

reprodução foi atingida quando a precisão entre dois valores são maiores ou iguais a 95%. Porém, é necessário que o cientista avalie se a reprodução obtida está dentro dos parâmetros aceitáveis.

A falta de disponibilidade de componentes externos, tais como serviços *web* pode limitar a reprodução. Se um *workflow* adota um serviço *web* em alguma atividade, a reprodução desse *workflow* pode ser ameaçada por alterações ou indisponibilidade deste serviço. Para contornar este problema é possível usar os dados intermediários gerados na execução original no momento da reprodução. Se uma atividade está vinculada a um serviço *web* é possível o resultado original gerou um conjunto No entanto, é necessário analisar se a atividade que utiliza um serviço possui dependência de dados de atividades predecessoras. Caso exista, é preciso verificar se os dados que seriam usados como entrada da atividade com o serviço na reprodução gerou o mesmo conjunto de dados de entrada para essa mesma atividade no momento da produção dos dados. Caso os dados sejam semelhantes, podem ser considerados os dados intermediários gerados por esta atividade, que estão armazenados na proveniência, sem que seja necessário reexecutar o serviço. Porém, se os dados de entrada forem divergentes, a reprodução de um serviço pode ser comprometido

Para publicar a pesquisa reprodutível com todos os obsjetos de pesquisa, é necessário transferir toda a infraestrutura de dados para o provedor de nuvem selecionado. Portanto, a viabilidade da movimentação dos dados para os provedores de nuvem pode ser dificultada devido ao tamanho dos conjuntos de dados dos experimento. É importante ressaltar que alguns provedores de computação em nuvem, como a *Amazon Web Service*[3] estão optando pelo armazenamento de grandes bases de dados científicas, sendo um fator que motiva a implantação dos experimentos na nuvem, migrando a computação para o local onde os dados estão armazenado.

5.7 Considerações Finais

A abordagem ReproeScience foi concebida para atender a um conjunto de variáveis relacionadas com portabilidade do experimento de um ambiente operacional de computação de alto desempenho para a infraestrutura de provedores de nuvem, analisando o nível de transparência de uma pesquisa, tendo como meta um maior nível de compartilhamento dos artefatos usados para a produção dos resultados para o público em geral, e ao final, avaliar o quanto é possível reproduzir cada um desses elementos de uma pesquisa.

Os metadados e dados de proveniência são informações importantes para auxiliar as análises das trilhas de execução de um experimento. Mesmo que em alguns casos não seja possível mostrar com exatidão a reprodução total da execução de

[3]https://aws.amazon.com/datasets/

um *workflow*, é possível avaliar o quanto os objetos de pesquisa de um experimento computacional podem fielmente ser reproduzidos e quais são as variáveis importantes para identificar essa reprodução. A adoção da nuvem não permite apenas a reprodução dos resultados em termos de objetos de pesquisa, mas também garante recursos necessários para a obtenção da mesma taxa de desempenho alcançada na execução do experimento original. Foi observado que cada experimento tem demandas próprias de recursos, tais como processamento e armazenamento, e que é possível obter os recursos necessários nos provedores de nuvem para o atendimento dessa demanda.

Todo o processo de reprodução científica deve ser passível de validação e verificação. É preciso verificar se a metodologia proposta foi devidamente reimplantada e reproduzida e se a reprodução alcançou os resultados equivalentes ao obtidos com o experimento original consolidado. Foi mostrado que os resultados produzidos por uma pesquisa podem ser exatos ou aproximados. Quando os valores são exatos, independente do número de vezes que o experimento for executado, os resultados obtidos terão valores equivalentes. Entretanto, no caso dos valores aleatórios, sempre existirá a possibilidade de duas diferentes execuções produzirem valores diferentes. Pode ocorrer de experimentos que geram resultados não determinístico produzirem valores equivalentes, porém, isso não é garantido. É preciso avaliar nesses casos a precisão entre os resultados para verificar o intervalo de diferença do resultado da reprodução em relação a produção dos resultados.

A reprodutibilidade trabalha com a possibilidade de alteração do ambiente experimental, de forma os métodos sejam robustos a essas alterações. No caso os estudos de casos, foi verificada com o *workflow* SciCumulus, a robustez mediante a alteração na infraestrutura, ou seja, mesmo mudando o ambiente de execução os resultados de proveniência mantiveram-se os mesmos. No caso do *workflow* Montage, as mudanças no ambiente afetaram as equivalências nos resultados intermediários, porém, o resultado final obtido (imagem do espaço em JPEG) foi visualmente similar. O *workflow* pode ser sensível a precisão dos cálculos aritméticos, porém, a morfologia do grafo permaneceu equivalente, entretanto, foi obtido o mesmo número de vértices e arestas (arestas são bastante específicas *WasGeneratedBy* e *Used*, por exemplo), caracterizando o grafo de proveniência como isomórfico.

Capítulo 6

Conclusões

Este trabalho discutiu a importância da ciência da computação como peça fundamental para o desenvolvimento da ciência nos mais variados campos de investigação científica. A computação pode prover a capacidade necessária para tornar a ciência mais reprodutível acelerando a produção dos cientistas e a concepção de novas descobertas. Um dos grandes desafios concentrava-se nas barreiras técnicas de preservação do ambiente operacional a longo prazo, assim como as questões relacionadas com a portabilidade da pesquisa para uma infraestrutura computacional diferente daquela responsável pela produção dos resultados. No entanto, a abordagem ReproeScience superou tais desafios aplicando a virtualização e a computação em nuvem para preservar o ambiente operacional permitindo o acesso aos recursos necessários para a reprodução. Além disso, as técnicas de monitoramento do ambiente unidas com a captura e armazenamento de metadados e dados de proveniência foram essenciais para os processos de reimplantação e reprodução do ambiente operacional. Os resultados dos experimentos confirmaram a reprodução dos estudos de caso e ainda mostraram algumas peculiaridades a serem tratadas por métodos de verificação e validação dos resultados científicos.

Foram propostos mecanismos de monitoramento para a coleta das características e comportamento da execução dos experimentos científicos computacionais. Este monitoramento permitiu a identificação dos parâmetros, dados, arquivos, infraestrutura, usuários e demais objetos de pesquisa usados em tempo de execução. A representação desse conjunto de elementos conforme os modelos e padrões de proveniência permitiu agregar o valor de confiabilidade aos objetos de pesquisa devido à possibilidade de rastreamento da origem de cada elemento. A adoção da proviência permitiu a construção do mecanismo de análise da reprodutibilidade evidenciando os pontos de equivalência e divergência na produção e reprodução dos resultados. Esse mecanismo mostrou como a reprodução dos resultados pode ser obtida mesmo portando os experimentos do ambiente computacional para os provedores de nuvem pública.

A definição das duas estratégias de monitoramento adotadas pela abordagem auxiliaram a consolidar os resultados obtidos, pois os resultados gerados por uma validaram os obtidos pela outra. Ambos monitores alcançaram resultados bastante similares. Essas informações reforçam a hipótese de que até mesmo diferentes trilhas de execução podem gerar um resultado final equivalente, respeitando o uso dos mesmos conjuntos de dados de entrada, parâmetros e sequência de execução. Os resultados mostraram altos índices de equivalência no estudo de caso intensivo de processamento, em alguns casos alcançando o isomorfismo na trilha de execução. No estudo de caso intesivo de dados foi evidenciado que as questões relacionadas a precisão dos sistemas de computadores podem diminuir a precisão dos resultados. Resultados obtidos em um ambiente de produção podem ter resultados com maior precisão em relação aos obtidos em um ambiente de reprodução, neste caso, nos provedores de nuvem. Mesmo a taxa de equivalência sendo menor, comparando os resultados produzidos em relação aos reproduzidos, foi verificado que o produto final das reproduções foram bastante equivalentes e as diferenças imperceptíveis.

A abordagem adotou a definição de pacotes de pesquisa reprodutíveis contendo todos os objetos de pesquisa armazenados em um diretório específico para a concentração dos elementos usados e gerados no experimento em tempo de execução. Portanto, este pacote concentra todos os elementos necessários para a reimplantação da infraestrutura e reprodução dos resultados produzidos pela pesquisa. Ele permite ainda a compactação dos arquivos, programas, bibliotecas e demais elementos para uma transferência mais eficiente para infraestruturas externas através da rede.

A reimplantação da infraestrutura computacional para a reprodução apresentou desafios em relação aos recursos de computação, comunicação em rede de computadores e, principalmente, em termos de armazenamento dos dados. Os testes mostraram que os experimentos que demandam por apenas um nó de processamento permitem uma reimplantação transparente e não afetam a reexecução e reprodução dos resultados. Entretanto, a reimplantação de infraestruturas que usam dois ou mais nós abrem desafios relacionados com a comunicação na rede, e ainda problemas relacionados com o armazenamento distribuído dos dados. Esses desafios foram superados na abordagem por meio da adoção de sistemas de arquivos distribuídos na infraestrutura da nuvem viabilizando a reexecução dos experimentos.

O conjunto de definições e termos relacionados com a reprodutibilidade científica na *e-Science* levantados na bibliografia abriram uma oportunidade para a realização de uma classificação desses conceitos em uma estrutura de taxonomia. Foi mostrado que essa taxonomia pode auxiliar na identificação dos conceitos relacionados, e ainda, mostrar onde eles se encaixam na estrutura como um todo. A taxonomia proposta forneceu condições para a construção de estrutura para orientar a comparação entre abordagens e sistemas construídos para esse propósito.

Foi apresentado o resultado da compilação de diversos esforços e soluções aplicadas na área de pesquisa reprodutível. Este estudo resultou na apresentação do atual estado da arte sobre o tema, caracterizando as abordagens e comparando suas principais características em relação à proposta de abordagem de reprodutibilidade apresentada nesta tese. Diante do cenário, foi possível aplicar a taxonomia proposta para a construção de um mecanismo para a comparação das abordagens levantadas na bibliografia. O resultado do levantamento bibliográfico e do estado da arte foi representado por uma arquitetura de referência para a concepção de soluções para este fim, baseada nas diversas soluções propostas e desafios enfrentados.

Apesar da motivação do desenvolvimento de uma pesquisa reprodutível ser a transparência, através da validação e verificação dos processos e resultados publicados, existem ainda diversas questões que precisam ser tratadas e resolvidas. Foram mostradas que a validação e verificação são apenas dois dos diversos requisitos relacionados a este tema. Estão vinculadas as questões de apresentação e hospedagem de artigos executáveis, edição e construção de referências a objetos de pesquisa, gestão de autoria, licenciamento, direito de cópia e distribuição dos objetos de pesquisa, controle de versão, identificação única, gestão de documentação e anotação, além da obtenção de toda a infraestrutura necessária para a reprodução de uma pesquisa.

Portanto, a abordagem ReproeScience apresentou uma proposta para simplificar o processo de criação de pesquisas reprodutíveis. Além disso, apresentou um método para apoiar os processos de validação e verificação dos resultados por meio da análise dos dados de proveniência de um experimento científico computacional. A adoção dos provedores de computação em nuvem permitiu a criação de um repositório para a publicação e compartilhamento dos objetos de pesquisa usados na derivação dos resultados científicos e para a construção de um artigo científico digital.

6.1 Trabalhos Futuros

É recomendado que os cientistas associem atividades que ajudam outros usuários a testar os resultados produzidos, pois cada área de aplicação avalia tipos específicos de dados e arquivos, dificultando a criação de um método genérico de verificação dos resultados. Portanto, ao definir um experimento computacional é necessário criar uma atividade específica para a verificação analítica dos resultados. Grande parte das abordagens de reprodução permitem apenas a avaliação dos dados derivados, de forma visual, a partir de mecanismos de apresentação baseados em gráficos, imagens e vídeos. Apesar da análise da proveniência proposta nesta tese ser um excelente meio de avaliação da reprodução é necessário buscar formas genéricas de avaliar o resultado final produzido pelo *workflow* como uma forma de adicionar maior confiabilidade aos resultados.

Atualmente, as publicações científicas possuem um modelo de identificação de objetos do tipo documentos denominado DOI (*Document Object Identifier*), que faz referência univocamente a uma publicação. No caso dos objetos científicos, usados como em um experimento, tal abordagem tornará necessária, principalmente frente as modificações em conjuntos de dados usados em ensaios, inclusão e/ou retirada de determinadas etapas de um experimento, modificações em *softwares* e bibliotecas, ou seja, a geração de novas versões, bem como a reutilização de partes do experimento em outras pesquisas. Portanto, é necessário criar uma estrutura de identificação única para cada objeto de pesquisa associado a uma publicação científica de forma que seja adotada e reconhecida pela comunidade científica como um todo.

Os testes mostraram a necessidade de tornar as questões relacionadas com a interoperabilidade entre os provedores de nuvem mais transparentes. O ReproeScience permite fazer a migração dos objetos de pesquisa de uma infraestrutura de nuvem para a outra de forma transparente por meio dos pacotes reprodutíveis. Entretanto, seria mais conveniente a possibilidade de portabilidade da infraestrutura em nível de VM. Essa capacidade permitiria a produção de mecanismos de seleção das melhores ofertas ou fornecimento de serviços baseados em melhores QoS (desempenho, vazão de rede, volume de armazenamento etc).

6.2 Produção bibliográfica

Este trabalho resultou na publicação de um artigo no Simpósio Brasileiro de Banco de Dados (2012) intitulado *Reprodução de Experimentos Científicos Usando Nuvens*[1], de autoria de Ary H M Oliveira, Murilo Martins, Igor Modesto, Daniel de Oliveira e Marta Mattoso, assim como a aprovação de um tutorial para ser apresentado no Simpósio Brasileiro de Banco de Dados (2015), de autoria de Ary H. M. Oliveira, Daniel de Oliveira e Marta Mattoso, cujo título é *Reprodução de Experimentos Científicos: Teoria e Prática*.

Foi gerada a submissão de um artigo para o periódico Computing[2] da editora Springer, com autoria Ary H M Oliveira, Daniel de Oliveira e Marta Mattoso, intitulado *Reproducibility in Computational Science: a Systematic Review*. Além disso, está sendo produzido um artigo para ser submetido ao periódico *Simulation Modelling Practice and Theory*[3] da editora Elsevier com a autoria de Ary H. M. Oliveira, Daniel de Oliveira, Vitor Silva, Kary A. C. S. Ocaña e Marta Mattoso, o qual terá o título *Reproducing Scientific Workflows in Cloud Computing*.

[1] http://www.lbd.dcc.ufmg.br/colecoes/sbbd/2012/002.pdf
[2] http://www.springer.com/computer/journal/607
[3] http://www.journals.elsevier.com/simulation-modelling-practice-and-theory

Referências Bibliográficas

[1] MACKO, P., CHIARINI, M., SELTZER, M. "Collecting Provenance Via the Xen Hypervisor", *Proceedings of 3rd USENIX Workshop on the Theory and Practice of Provenance (TaPP '11)*, pp. 1–15, June 2011.

[2] CHIRIGATI, F., SHASHA, D., FREIRE, J. "Packing Experiments for Sharing and Publication". In: *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, SIGMOD '13, pp. 977–980, New York, NY, USA, 2013. ACM. ISBN: 978-1-4503-2037-5. doi: 10.1145/2463676.2465269. Disponível em: <http://doi.acm.org/10.1145/2463676.2465269>.

[3] NOWAKOWSKI, P., CIEPIELA, E., HAREZLAK, D., KOCOT, J., KASZTELNIK, M., BARTYNSKI, T., MEIZNER, J., DYK, G., MALAWSKI, M. "The Collage Authoring Environment", *Executable Paper Grand Challenge International Conference on Computational Science, ICCS 2011*, , n. 4, pp. 608–617, 2011.

[4] BRAMMER, G. R., CROSBY, R. W., MATTHEWS, S. J., WILLIAMS, T. L. "Paper Mâché: Creating Dynamic Reproducible Science", *Procedia Computer Science*, v. 4, n. 0, pp. 658 – 667, 2011. ISSN: 1877-0509. doi: http://dx.doi.org/10.1016/j.procs.2011.04.069. Disponível em: <http://www.sciencedirect.com/science/article/pii/S187705091100127X>. Proceedings of the International Conference on Computational Science, ICCS 2011.

[5] FOMEL, S., HENNENFENT, G. "Reproducible Computational Experiments Using SCONS". In: *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2007)*, v. 4, pp. IV–1257–IV–1260, April 2007. doi: 10.1109/ICASSP.2007.367305.

[6] STRIJKERS, R., CUSHING, R., VASYUNIN, D., DE LAAT, C., BELLOUM, A. S. Z., MEIJER, R. "Toward Executable Scientific Publications", *Procedia Computer Science*, v. 4, pp. 707–715, 2011. ISSN: 1877-0509.

[7] OCAÑA, K. A., DE OLIVEIRA, D., HORTA, F., DIAS, J., OGASAWARA, E., MATTOSO, M. "Exploring Molecular Evolution Reconstruction Using a Parallel Cloud Based Scientific Workflow". In: *Proceedings of the 7th Brazilian Symposium on Bioinformatics - BSB 2012*, n. 7409, pp. 179–191. Springer-Verlag, August 2012.

[8] MOREAU, L., CLIFFORD, B., FREIRE, J., FUTRELLE, J., GIL, Y., GROTH, P., KWASNIKOWSKA, N., MILES, S., PLALE, B., SIMMAHAN, Y., STEPHAN, E., DEN BUSSCHE, J. V. "The Open Provenance Model Core Specification (v1.1)", *Future Generation Computer Systems*, v. 6, n. 27, pp. 743–756, June 2011.

[9] "W3C Prov: Prov Overview". March 2014. Disponível em: <http://www.w3.org/TR/2013/NOTE-prov-overview-20130430/>.

[10] "PROV-DM: The PROV Data Model". April 2013. Disponível em: <http://www.w3.org/TR/prov-dm/>.

[11] "Executable Paper Grand Challenge". June 2011. Disponível em: <http://www.executablepapers.com/index.html>.

[12] "Montage: An Astronomical Image Mosaic Engine". November 2014. Disponível em: <http://montage.ipac.caltech.edu/index.html>.

[13] SZALAY, A. S., BLAKELEY, J. A. "Gray's laws: database-centric computing in science." In: Hey, T., Tansley, S., Tolle, K. M. (Eds.), *The Fourth Paradigm*, Microsoft Research, pp. 5–11, 2009. ISBN: 978-0982544204. Disponível em: <http://dblp.uni-trier.de/db/books/collections/4paradigm2009.html#SzalayB09>.

[14] KARPATHIOTAKIS, M., DE OLIVEIRA BRANCO, M. S., ALAGIANNIS, I., AILAMAKI, A. "Adaptive Query Processing on RAW Data". In: *40th International Conference on Very Large Databases*, 2014.

[15] JUVE, G., DEELMAN, E. "Scientific Workflows and Clouds", *Crossroads*, v. 16, n. 3, pp. 14–18, mar. 2010. ISSN: 1528-4972. doi: 10.1145/1734160.1734166. Disponível em: <http://doi.acm.org/10.1145/1734160.1734166>.

[16] BERRIMAN, G. B., G. J. C. C. D. W. J. J. C. K. D. S. P. T. A. "Montage: An On-Demand Image Mosaic Service for the NVO", *Astronomical Data Analysis Software and Systems XII ASP Conference Series*, v. 295, pp. 343–346, 2003.

[17] "Jim Gray on eScience: A Transformed Scientific Method". In: Hey, T., Tansley, S., Tolle, K. (Eds.), *The Fourth Paradigm: Data-Intensive Scientific Discovery*, Microsoft Research, 2009. Disponível em: <http://research.microsoft.com/en-us/collaboration/fourthparadigm/4th_paradigm_book_jim_gray_transcript.pdf>.

[18] "Defining e-Science - National e-Science Centre". July 2014. Disponível em: <http://www.nesc.ac.uk/nesc/define.html>.

[19] OCAÑA, K. A. C. S., DE OLIVEIRA, D., OGASAWARA, E., DÁVILA, A. M. R., LIMA, A. B. A., MATTOSO, M. "SciPhy: A Cloud-Based Workflow for Phylogenetic Analysis of Drugs Target in Protozoan Genomes". In: O. Norberto de Souza, G. P. T., Palakal, M. J. (Eds.), *Advances in Bioinformatics and Computational Biology*, pp. 66–70, Springer-Verlag, 2011.

[20] DEELMAN, E., GANNON, D., SHIELDS, M., TAYLOR, I. "Workflows and e-Science: An Overview of Workflow System Features and Capabilities", *Future Gener. Comput. Syst.*, v. 25, n. 5, pp. 528–540, maio 2009. ISSN: 0167-739X. doi: 10.1016/j.future.2008.06.012. Disponível em: <http://dx.doi.org/10.1016/j.future.2008.06.012>.

[21] MESIROV, J. P. "Accessible Reproducible Research", *Science*, v. 327, n. 5964, pp. 415–416, 2010.

[22] GOBLE, C. "The Reality of Reproducibility in Computational Science: reproduce? repeat? rerun? and does it matter. Keynotes and Panels", *8th IEEE International Conference on eScience*, v. 327, n. 5964, pp. 415–416, October 2012.

[23] KLINGINSMITH, J., MAHOUI, M., WU, Y. "Towards Reproducible eScience in the Cloud". In: *Cloud Computing Technology and Science (CloudCom), 2011 IEEE Third International Conference on*, pp. 582–586, Nov 2011. doi: 10.1109/CloudCom.2011.89.

[24] ARMBRUST, M., FOX, A., GRIFFITH, R., JOSEPH, A. D., KATZ, R., KONWINSKI, A., LEE, G., PATTERSON, D., RABKIN, A., STOICA, I., ZAHARIA, M. "A View of Cloud Computing", *Commun. ACM*, v. 53, n. 4, pp. 50–58, abr. 2010. ISSN: 0001-0782. doi: 10.1145/1721654.1721672. Disponível em: <http://doi.acm.org/10.1145/1721654.1721672>.

[25] MELL, P. M., GRANCE, T. *SP 800-145. The NIST Definition of Cloud Computing*. Relatório técnico, Gaithersburg, MD, United States, 2011.

[26] "Cloud Infrastructure and Services Student Guide", *EMC Education Services*, , n. 111711, November 2011.

[27] VAQUERO, L. M., RODERO-MERINO, L., CÁCERES, J., LINDNER, M. "A Break in the Clouds: Towards a Cloud Definition", *ACM SIGCOMM Computer Communication Review*, v. 39, n. 1, pp. 32–35, January 2009.

[28] KEARNEY, K. T., TORELLI, F. "The SLA Model". In: Philipp Wieder, Joe M. Butler, W. T., Yahyapour, R. (Eds.), *Service Level Agreements for Cloud Computing*, v. 1, pp. 43–68, Springer, 2012.

[29] SMITH, J., NAIR, R. *Virtual Machines: Versatile Platforms for Systems and Processes (The Morgan Kaufmann Series in Computer Architecture and Design)*. San Francisco, CA, USA, Morgan Kaufmann Publishers Inc., 2005. ISBN: 1558609105.

[30] PAPAZOGLOU, M., VAN DEN HEUVEL, W.-J. "Service oriented architectures: approaches, technologies and research issues", *The VLDB Journal*, v. 16, n. 3, pp. 389–415, 2007. ISSN: 1066-8888. doi: 10.1007/s00778-007-0044-3. Disponível em: <http://dx.doi.org/10.1007/s00778-007-0044-3>.

[31] KOOP, D., SANTOS, E., MATES, P., VO, H. T., BONNET, P., BAUER, B., SURER, B., TROYER, M., WILLIAMS, D. N., TOHLINE, J. E., FREIRE, J., SILVA, C. T. "A Provenance-Based Infrastructure to Support the Life Cycle of Executable Papers", *Procedia Computer Science*, v. 4, n. 0, pp. 648 – 657, 2011. ISSN: 1877-0509. doi: http://dx.doi.org/10.1016/j.procs.2011.04.068. Disponível em: <http://www.sciencedirect.com/science/article/pii/S1877050911001268>. Proceedings of the International Conference on Computational Science, {ICCS} 2011.

[32] DE OLIVEIRA, D., OGASAWARA, E., BAIÃO, F., MATTOSO, M. "SciCumulus: A Lightweigh Cloud Middleware to Explore Many Task Computing Paradigm in Scientific Workflows", *Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on Cloud Computing*, pp. 378–385, 2010.

[33] STODDEN, V., DONOHO, D., FOMEL, S., FREIDLANDER, M. P., GERSTEIN, M., LEVEQUE, R., MITCHELL, I., LARRIMORE OUELLETTE,

L., WIGGINS, C. “Reproducible Research: Addressing the Need for Data and Code Sharing in Computational Science”, *Computing in Science and Engineering*, v. 12, n. 5, pp. 8–13, set. 2010. ISSN: 1521-9615. doi: 10.1109/mcse.2010.113. Disponível em: <http://dx.doi.org/10.1109/mcse.2010.113>.

[34] BRIAN MATTHEWS, ARIF SHAON, J. B. C. J. “A Framework for Software Preservation”, *International Journal of Digital Curation*, v. 5, n. 1, pp. 91–105, 2010. doi: doi:10.2218/ijdc.v5i1.145.

[35] ABADI, D., AGRAWAL, R., AILAMAKI, A., BALAZINSKA, M., BERNSTEIN, P. A., CAREY, M. J., CHAUDHURI, S., DEAN, J., DOAN, A., FRANKLIN, M. J., GEHRKE, J., HAAS, L. M., HALEVY, A. Y., HELLERSTEIN, J. M., IOANNIDIS, Y. E., JAGADISH, H. V., KOSSMANN, D., MADDEN, S., MEHROTRA, S., MILO, T., NAUGHTON, J. F., RAMAKRISHNAN, R., MARKL, V., OLSTON, C., OOI, B. C., RÉ, C., SUCIU, D., STONEBRAKER, M., WALTER, T., WIDOM, J. “The Beckman Report on Database Research”, *SIGMOD Rec.*, v. 43, n. 3, pp. 61–70, dez. 2014. ISSN: 0163-5808. doi: 10.1145/2694428.2694441. Disponível em: <http://doi.acm.org/10.1145/2694428.2694441>.

[36] VANDEWALLE, P. “Code Sharing Is Associated with Research Impact in Image Processing”, *Computing in Science & Engineering*, pp. 42–47, July/August 2012. ISSN: 0362-1340.

[37] LEVEQUE, R. J. “Python Tools for Reproducible Research on Hyperbolic Problems”, *Computing in Science Engineering*, v. 11, n. 1, pp. 19–27, Jan 2009. ISSN: 1521-9615. doi: 10.1109/MCSE.2009.13.

[38] DUDLEY, J. T., BUTTE, A. J. “In silico research in the era of cloud computing”, *Nature Biotechnology*, v. 28, n. 11, pp. 1181—-185, 2010. ISSN: 1087-0156. doi: 10.1038/nbt1110-1181. Disponível em: <http://dx.doi.org/10.1038/nbt1110-1181>.

[39] HINSEN, K. “A data and code model for reproducible research and executable”, *Procedia Computer Science*, v. 4, n. 0, pp. 579–588, 2011. Proceedings of the International Conference on Computational Science, ICCS 2011.

[40] KRISHNAMURTHI, S., VITEK, J. “The Real Software Crisis: Repeatability As a Core Value”, *Commun. ACM*, v. 58, n. 3, pp. 34–36, fev. 2015. ISSN: 0001-0782. doi: 10.1145/2658987. Disponível em: <http://doi.acm.org/10.1145/2658987>.

[41] HUANG, Y., GOTTARDO, R. "Comparability and reproducibility of biomedical data", *Briefings in Bioinformatics*, v. 14, n. 4, pp. 391–401, 2013. doi: 10.1093/bib/bbs078. Disponível em: <http://bib.oxfordjournals.org/content/14/4/391.abstract>.

[42] MCNUTT, M. "Journals unite for reproducibility", *Science*, v. 346, n. 6210, pp. 679, nov. 2014. doi: DOI:10.1126/science.aaa1724. Disponível em: <http://www.sciencemag.org/content/346/6210/679.full>.

[43] VANDEWALLE, P., BARRENETXEA, G., JOVANOVIC, I., RIDOLFI, A., VETTERLI, M. "Experiences with Reproducible Research in Various Facets of Signal Processing Research". In: *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, v. 4, pp. IV–1253–IV–1256, April 2007.

[44] DONOHO, D. L. "An invitation to reproducible computational research", *Biostatistics*, v. 3, n. 11, pp. 376–388, 2010.

[45] *Guide to the use of repeatability, reproducibility and trueness estimates in measurement uncertainty estimation*. In: Technical Report ISO/DTS 21748 ISO/TC 69/SC 6 N 456, International Organization for Standardization, January 2002.

[46] FOMEL, S., CLAERBOUT, J. F. "Reproducible Research", *Computing in Science & Engineering*, pp. 5–7, January/February 2009.

[47] CASADEVALL, A., FANG, F. C. "Reproducible Science", *Infection and Immunity*, v. 12, n. 78, pp. 4972–4975, December 2010.

[48] FREIRE, J., BONNET, P., SHASHA, D. "Computational Reproducibility: State-of-the-art, Challenges, and Database Research Opportunities". In: *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, SIGMOD '12, pp. 593–596, New York, NY, USA, 2012. ACM. ISBN: 978-1-4503-1247-9. doi: 10.1145/2213836.2213908. Disponível em: <http://doi.acm.org/10.1145/2213836.2213908>.

[49] STODDEN, V., BAILEY, D. H., BORWEIN, J., LEVEQUE, R. J., RIDER, W., STEIN, W. *Setting the Default to Reproducible: Reproducibility in Computational and Experimental Mathematics*. Relatório técnico, ICERM Workshop Reproducibility in computational and Experimental Mathematics, 2013.

[50] BELHAJJAME, K., ROURE, D. D., GOBLE, C. A. "Research Object Management: Opportunities and Challenges". February 2012.

[51] KOVACEVIC, J. "How to Encourage and Publish Reproducible Research". In: *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, v. 4, pp. IV–1273–IV–1276. IEEE, abr. 2007. ISBN: 1-4244-0727-3. doi: 10.1109/icassp.2007.367309. Disponível em: <http://dx.doi.org/10.1109/icassp.2007.367309>.

[52] DAVIDSON, S. B., FREIRE, J. "Provenance and Scientific Workflows: Challenges and Opportunities". In: *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, pp. 1345–1350, New York, NY, USA, 2008. ACM. ISBN: 978-1-60558-102-6. doi: 10.1145/1376616.1376772. Disponível em: <http://doi.acm.org/10.1145/1376616.1376772>.

[53] SCHWAB, M., KARRENBACH, M., CLAERBOUT, J. "Making Scientific Computations Reproducible", *Computing in Science and Engg.*, v. 2, n. 6, pp. 61–67, nov. 2000. ISSN: 1521-9615. doi: 10.1109/5992.881708. Disponível em: <http://dx.doi.org/10.1109/5992.881708>.

[54] BAGGERLY, K. A., BERRY, D. A. "Reproducible Research", *Amstatnews: The Membership Magazine of the American Statistical Association*, 2012. Disponível em: <http://doi.acm.org/10.1145/2034863.2034873>.

[55] HEATHER A. PIWOWAR, R. S. D., FRIDSMA, D. B. "Sharing Detailed Research Data Is Associated with Increased Citation Rate Heather", mar. 2007. doi: 10.1371/journal.pone.0000308.

[56] MATTHEWS, J. N. "The Case for Repeated Research in Operating Systems", *SIGOPS Oper. Syst. Rev.*, v. 38, n. 2, pp. 5–7, abr. 2004. ISSN: 0163-5980. doi: 10.1145/991130.991131. Disponível em: <http://doi.acm.org/10.1145/991130.991131>.

[57] LEVEQUE, R. J., MITCHELL, I. M., STODDEN, V. "Reproducible research for scientific computing: Tools and strategies for changing the culture", *Computing in Science and Engineering*, v. 14, n. 4, pp. 13–17, 2012. ISSN: 1521-9615. doi: http://doi.ieeecomputersociety.org/10.1109/MCSE.2012.38.

[58] LOSCALZO, J. "Irreproducible experimental results: causes, (mis)interpretations, and consequences", *Educational Researcher*, v. 10, n. 125, pp. 1211–1214, Mar 2012. doi: 10.1161/CIRCULATIONAHA.112.098244.

[59] VITEK, J., KALIBERA, T. "R3: Repeatability, Reproducibility and Rigor", *SIGPLAN Not.*, v. 47, n. 4a, pp. 30–36, mar. 2012. ISSN: 0362-1340. doi: 10.1145/2442776.2442781. Disponível em: <http://doi.acm.org/10.1145/2442776.2442781>.

[60] KEIDING, N. "Reproducible research and the substantive context", *Biostatistics*, v. 11, n. 3, pp. 376–8, 2010. ISSN: 1465-4644. doi: 10.1093/biostatistics/kxq033.

[61] FOMEL, S. "Reproducible Research as a Community Effort: Lessons from the Madagascar Project", *Computing in Science & Engineering*, pp. 20–26, January/February 2015.

[62] "Madagascar". January 2015. Disponível em: <http://www.ahay.org/wiki/Main_Page>.

[63] BAKER, M. "Independent labs to verify high-profile papers", *Nature*, ago. 2012. ISSN: 1476-4687. doi: 10.1038/nature.2012.11176. Disponível em: <http://dx.doi.org/10.1038/nature.2012.11176>.

[64] HEROUX, M. A. "Improving CSE Software Through Reproducibility Requirements". In: *Proceedings of the 4th International Workshop on Software Engineering for Computational Science and Engineering*, SECSE '11, pp. 28–31, New York, NY, USA, 2011. ACM. ISBN: 978-1-4503-0598-3. doi: 10.1145/1985782.1985787. Disponível em: <http://doi.acm.org/10.1145/1985782.1985787>.

[65] DONOHO, D., MALEKI, A., RAHMAN, I., SHAHRAM, M., STODDEN, V. "Reproducible Research in Computational Harmonic Analysis", *Computing in Science Engineering*, v. 11, n. 1, pp. 8–18, Jan 2009. ISSN: 1521-9615. doi: 10.1109/MCSE.2009.15.

[66] STODDEN, V. "The Legal Framework for Reproducible Scientific Research: Licensing and Copyright", *Computing in Science & Engineering*, v. 11, n. 1, pp. 35–40, jan. 2009. ISSN: 1521-9615. doi: 10.1109/mcse.2009.19. Disponível em: <http://dx.doi.org/10.1109/mcse.2009.19>.

[67] BONNET, P., MANEGOLD, S., BJØRLING, M., CAO, W., GONZALEZ, J., GRANADOS, J., HALL, N., IDREOS, S., IVANOVA, M., JOHNSON, R., KOOP, D., KRASKA, T., MÜLLER, R., OLTEANU, D., PAPOTTI, P., REILLY, C., TSIROGIANNIS, D., YU, C., FREIRE, J., SHASHA, D. "Repeatability and Workability Evaluation of SIGMOD 2011", *SIGMOD Rec.*, v. 40, n. 2, pp. 45–48, set. 2011. ISSN: 0163-5808. doi:

10.1145/2034863.2034873. Disponível em: <http://doi.acm.org/10.1145/2034863.2034873>.

[68] "Reproducibility Initiative: Validation by Science Exchange Network". May 2014. Disponível em: <http://validation.scienceexchange.com/#/reproducibility-initiative>.

[69] "Nature Policies: Availability of data, material and methods". June 2014. Disponível em: <http://www.nature.com/authors/policies/availability.html>.

[70] "Science Information for Authors". Mar 2015. Disponível em: <http://www.sciencemag.org/site/feature/contribinfo/>.

[71] PENG, R. D. "Reproducible Research and Biostatistic (Editoral)", *Biostatistics*, v. 3, n. 10, pp. 405–408, 2009.

[72] LIN, Y.-H., KO, T.-M., CHUANG, T.-R., LIN, K.-J. "Open Source Licenses and the Creative Commons Framework: License Selection and Comparison", *Journal of Information Science and Engineering*, v. 22, n. 1, pp. 1–17, 2006. Disponível em: <http://www.iis.sinica.edu.tw/~{}trc/public/publications/jise06/jise06LinKoChuangLin.pdf>.

[73] GARCELON, M. "An information commons? Creative Commons and public access to cultural creations." *New Media & Society*, v. 8, n. 11, pp. 1307–1326, 2009. doi: 10.1186/gb-2004-5-10-r80.

[74] VÄLIMÄKI, M. "GNU General Public License and the Distribution of Derivative Works", *Journal of Information, Law and Technology*, v. 2005, n. 1, 2005.

[75] GAVISH, M., DONOHO, D. "A Universal Identifier for Computational Results", *Procedia Computer Science*, v. 4, n. 0, pp. 637 – 647, 2011. ISSN: 1877-0509. doi: http://dx.doi.org/10.1016/j.procs.2011.04.067. Disponível em: <http://www.sciencedirect.com/science/article/pii/S1877050911001256>. Proceedings of the International Conference on Computational Science, {ICCS} 2011.

[76] MISSIER, P., WOODMAN, S., HIDEN, H., WATSON, P. "Provenance and data differencing for workflow reproducibility analysis", *Concurrency and Computation: Practice and Experience*, pp. n/a–n/a, 2013. ISSN: 1532-0634. doi: 10.1002/cpe.3035. Disponível em: <http://dx.doi.org/10.1002/cpe.3035>.

[77] GREENBERG, J. "Metadata and the World Wide Web". In: *Encyclopedia of Library and Information Science*, pp. 1876–1888. Marcel Dekker, 2003.

[78] DEELMAN, E., BERRIMAN, B., CHERVENAK, A., CORCHO, O., GROTH, P., MOREAU, L. "Metadata and Provenance Management". In: Shoshani, A., Rotem, D. (Eds.), *Scientific Data Management: Challenges, Technology and Deployment*, Chapman & Hall/CRC, 2010.

[79] MOREAU, L., GROTH, P. T. *Provenance: An Introduction to PROV*. Synthesis Lectures on the Semantic Web: Theory and Technology. Morgan & Claypool Publishers, 2013. ISBN: 9781627052221.

[80] SIMMHAN, Y. L., PLALE, B., GANNON, D. "A Survey of Data Provenance in e-Science", *SIGMOD Rec.*, v. 34, n. 3, pp. 31–36, set. 2005. ISSN: 0163-5808. doi: 10.1145/1084805.1084812. Disponível em: <http://doi.acm.org/10.1145/1084805.1084812>.

[81] "BURRITO: Wrapping Your Lab Notebook in Computational Infrastructure". In: *Presented as part of the 4th USENIX Workshop on the Theory and Practice of Provenance*, Berkeley, CA, 2012. USENIX.

[82] TAYLOR, I., DEELMAN, E., GANNON, D. B., SHIELDS, M. *Workflows for e-Science: Scientific Worflows for Grids*. Springer, 2007.

[83] GUO, P. "CDE: A Tool for Creating Portable Experimental Software Packages", *Computing in Science and Engg.*, v. 14, n. 4, pp. 32–35, jul. 2012. ISSN: 1521-9615. doi: 10.1109/MCSE.2012.36. Disponível em: <http://dx.doi.org/10.1109/MCSE.2012.36>.

[84] MURTA, L., BRAGANHOLO, V., CHIRIGATI, F., KOOP, D., FREIRE, J. "noWorkflow: Capturing and Analyzing Provenance of Scripts". In: Ludäscher, B., Plale, B. (Eds.), *Provenance and Annotation of Data and Processes*, v. 8628, *Lecture Notes in Computer Science*, Springer International Publishing, pp. 71–83, 2015. ISBN: 978-3-319-16461-8. doi: 10.1007/978-3-319-16462-5_6. Disponível em: <http://dx.doi.org/10.1007/978-3-319-16462-5_6>.

[85] GUO, P. J., ENGLER, D. "CDE: Using System Call Interposition to Automatically Create Portable Software Packages". In: *Proceedings of the 2011 USENIX Conference on USENIX Annual Technical Conference*, USENIXATC'11, pp. 21–21, Berkeley, CA, USA, 2011. USENIX Association. Disponível em: <http://dl.acm.org/citation.cfm?id=2002181.2002202>.

[86] PIETER VAN GORP, S. M. "SHARE: a web portal for creating and sharing executable research papers", *International Conference on Computational Science, ICCS*, pp. 1–9, 2011.

[87] SICIAREK, J., WISZNIEWSKI, B. "IODA - an Interactive Open Document Architecture", *Executable Paper Grand Challenge International Conference on Computational Science, ICCS 2011*, , n. 4, pp. 668–677, 2011.

[88] EUGSTER, M. J. A., LEISCH, F. "Bench Plot and Mixed Effects Models: First steps toward a comprehensive benchmark analysis toolbox". In: Brito, P. (Ed.), *Compstat 2008—Proceedings in Computational Statistics*, pp. 299–306. Physica Verlag, Heidelberg, Germany, 2008. ISBN: 978-3-7908-2083-6.

[89] HOWE, B. "Virtual Appliances, Cloud Computing, and Reproducible Research", *Computing in Science and Engineering*, v. 14, n. 4, pp. 36–41, 2012. ISSN: 1521-9615. doi: http://doi.ieeecomputersociety.org/10.1109/MCSE.2012.62.

[90] DEELMAN, E., SINGH, G., LIVNY, M., BERRIMAN, B., GOOD, J. "The Cost of Doing Science on the Cloud: The Montage Example". In: *Proceedings of the 2008 ACM/IEEE Conference on Supercomputing*, SC '08, pp. 50:1–50:12, Piscataway, NJ, USA, 2008. IEEE Press. ISBN: 978-1-4244-2835-9. Disponível em: <http://dl.acm.org/citation.cfm?id=1413370.1413421>.

[91] OINN, T., ADDIS, M., FERRIS, J., MARVIN, D., SENGER, M., GREENWOOD, M., CARVER, T., GLOVER, K., POCOCK, M. R., WIPAT, A., LI, P. "Taverna: A Tool for the Composition and Enactment of Bioinformatics Workflows", *Bioinformatics*, v. 17, n. 20, pp. 3045–3054, 2004.

[92] FREIRE, J., SILVA, C. T., CALLAHAN, S. P., SANTOS, E., SCHEIDEGGER, C. E., VO, H. T. "Managing Rapidly-evolving Scientific Workflows". In: *Proceedings of the 2006 International Conference on Provenance and Annotation of Data*, IPAW'06, pp. 10–18, Berlin, Heidelberg, 2006. Springer-Verlag. ISBN: 3-540-46302-X, 978-3-540-46302-3. doi: 10.1007/11890850_2. Disponível em: <http://dx.doi.org/10.1007/11890850_2>.

[93] "PROV-XML: The PROV XML Schema". April 2013. Disponível em: <http://www.w3.org/TR/prov-xml/>.

[94] CHENEY, J., MISSIER, P., MOREAU, L. *Constraints of the PROV Data Model*. Technical report, {World Wide Web Consortium}, April 2013. Disponível em: <http://eprints.soton.ac.uk/356853/>.

[95] VON WINTERFELDT, D., FISCHER, G. "Multi-Attribute Utility Theory: Models and Assessment Procedures". In: Wendt, D., Vlek, C. (Eds.), *Utility, Probability, and Human Decision Making*, v. 11, *Theory and Decision Library*, Springer Netherlands, pp. 47–85, 1975. ISBN: 978-94-010-1836-4. doi: 10.1007/978-94-010-1834-0_3. Disponível em: <http://dx.doi.org/10.1007/978-94-010-1834-0_3>.

[96] MOSTÉFAOUI, G. K., KIM, M., CHUNG, M. "Supporting Adaptive Security Levels in Heterogeneous Environments". In: *Computational Science and Its Applications - ICCSA 2004, International Conference, Assisi, Italy, May 14-17, 2004, Proceedings, Part I*, pp. 537–546, 2004.

[97] MEIDANIS, J., SETUBAL, J. C. *Introduction to Computational Molecular Biology*. PWS Publisher, 1997.

[98] HOFFA, C., MEHTA, G., FREEMAN, T., DEELMAN, E., KEAHEY, K., BERRIMAN, B., GOOD, J. "On the Use of Cloud Computing for Scientific Workflows", *Fourth IEEE International Conference on eScience*, pp. 640–645, 2008.

[99] JUVE, G., DEELMAN, E., VAHI, K., MEHTA, G., BERMAN, B. P., BERRIMAN, B., MAECHLING, P. "Scientific Workflow Applications on Amazon EC2", *IEEE e-Science 2009 Workshops*, pp. 59–66, 2009.

[100] HIDEN, H., WOODMAN, S., WATSON, P., CALA, J. "Developing cloud applications using the e-Science Central platform", *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, v. 371, n. 1983, jan. 2013. ISSN: 1471-2962. doi: 10.1098/rsta.2012.0085. Disponível em: <http://dx.doi.org/10.1098/rsta.2012.0085>.

[101] COALITION, W. M. *Terminology & Glossary*. Relatório técnico, Technical Report WFMC-TC-1011, Issue 3.0, 1999. Disponível em: http://www.wfmc.org/docs/TC-1011_term_glossary_v3.pdf.

[102] LUDÄSCHER, B., ALTINTAS, I., BOWERS, S., CUMMINGS, J., CRITCHLOW, T., DEELMAN, E., ROURE, D. D., FREIRE, J., GOBLE, C., JONES, M., KLASKY, S., MCPHILLIPS, T., PODHORSZKI, N., SILVA,

C., TAYLOR, I., VOUK, M. "Scientific Process Automation and Workflow Management". In: Shoshani, A., Rotem, D. (Eds.), *Scientific Data Management*, Computational Science Series, Chapman & Hall, cap. 13, 2009.

[103] YANG, X., WALLOM, D., WADDINGTON, S., WANG, J., SHAON, A., MATTHEWS, B., WILSON, M., GUO, Y. G. L., BLOWER, J. D., VASILAKOS, A. V., LIU, K., KERSHAW, P. "Cloud Computing in e-Science: Research Challenges and Opportunities", v. 70, pp. 408–464, August 2014. doi: 10.1007/s11227-014-1251-5. Disponível em: <http://dx.doi.org/10.1007/s11227-014-1251-5>.

[104] FREIRE, J., KOOP, D., SANTOS, E., SILVA, C. T. "Provenance for Computational Tasks: A Survey", *Computing in Science and Engg.*, v. 10, n. 3, pp. 11–21, maio 2008. ISSN: 1521-9615. doi: 10.1109/MCSE.2008.79. Disponível em: <http://dx.doi.org/10.1109/MCSE.2008.79>.

[105] "Vistrails: An open-source scientific workflow and provenance management system that supports data exploration and visualization". July 2014. Disponível em: <http://www.vistrails.org/index.php/Main_Page>.

[106] CROWDLABS. "Home. crowdLabs". http://www.crowdlabs.org, 2015. Acessado em: 20/03/2015.

[107] BARGA, R. S., DIGIAMPIETRI, L. A. "Automatic Generation of Workflow Provenance". In: *Proceedings of the 2006 International Conference on Provenance and Annotation of Data*, IPAW'06, pp. 1–9, Berlin, Heidelberg, 2006. Springer-Verlag. ISBN: 3-540-46302-X, 978-3-540-46302-3. doi: 10.1007/11890850_1. Disponível em: <http://magazine.amstat.org/blog/2011/01/01/scipolicyjan11/>.

[108] MOREAU, L., LUDÄSCHER, B., ALTINTAS, I., BARGA, R. S., BOWERS, S., CALLAHAN, S., CHIN, G., CLIFFORD, B., COHEN, S., COHEN-BOULAKIA, S., OTHERS. "Special issue: The first provenance challenge", *Concurrency and computation: practice and experience*, v. 20, n. 5, pp. 409–418, 2008.

[109] "Provenance Challenge Wiki". 2014. Disponível em: <http://twiki.ipaw.info/bin/view/Challenge/>.

[110] ZHAO, Y., FEI, X., RAICU, I., LU, S. "Opportunities and Challenges in Running Scientific Workflows on the Cloud." In: *CyberC*, pp. 455–462.

IEEE, 2011. ISBN: 978-1-4577-1827-4. Disponível em: <http://dblp.uni-trier.de/db/conf/cyberc/cyberc2011.html#ZhaoFRL11>.

[111] WANG, L., VON LASZEWSKI, G., YOUNGE, A., HE, X., KUNZE, M., TAO, J., FU, C. "Cloud Computing: a Perspective Study", *New Generation Computing*, v. 28, n. 2, pp. 137–146, 2010. ISSN: 0288-3635. doi: 10.1007/s00354-008-0081-5. Disponível em: <http://dx.doi.org/10.1007/s00354-008-0081-5>.

Apêndice A

Infraestrututa para a Reprodução

Este apêndice apresenta os conceitos, métodos, ferramentas e tecnologias usados para construir a solução de reprodução descrita nos próximos capítulos. Eles servem como base para a concretização da abordagem proposta. São apresentados os contextos de aplicação de cada um desses conceitos, apontando as suas principais características e justificando a importância do uso. A adoção das tecnologias apresentadas é baseada na necessidade de atender a 4 requisitos para a composição da plataforma de experimentos computacionais [100]: (1) armazenamento, (2) analises, (3) automatização dos processos de análise, e (4) compartilhamento dos dados.

Este capítulo está organizado em 4 seções. A Seção A.1 apresenta os conceitos dos *workflows* científicos e Sistemas de Gerencia de *Workflows* aplicados como a base para a representação computacional dos experimentos computacionais. A Seção A.2 apresenta as definições dos metadados e proveniência como elementos importantes para representação das informações das execuções dos conjuntos de ensaios dos experimentos científicos. Ao final, a Seção A.3 apresenta a infraestrutura computacional sugerida para a reprodução dos experimentos apoiados pela abordagem proposta.

A.1 *Workflows* Científicos

Na realização de procedimentos científicos, os pesquisadores são guiados por sequências de tarefas, com uma ordem pré determinada de execução para a definição de um fluxo de trabalho. A representação desse fluxo, também denominado *workflow*, envolve a execução coordenada de várias tarefas realizadas por diferentes entidades de processamento. A *Workflow Management Coalition* [101] define um *worfklow* como a automatização de um processo de negócio, no todo ou em parte, durante a qual os documentos, informações ou tarefas são passadas de um participante para outro por uma ação, de acordo com um conjunto das normas processuais. Um *workflow* é uma abstração que permite compor um conjunto de programas e *scripts* na forma de fluxo de atividades com o objetivo de se atingir um resultado desejado

[82].

Ludäscher *et al.* [102] define um *workflow* científico como uma descrição formal de um processo para a realização de um objetivo científico, normalmente expresso em termos de tarefas e suas dependências. A composição de conjuntos de programas e *scripts* de um *workflow* é definido como orquestração [20], a qual é responsável pela definição da sequencia de tarefas necessárias para gerenciar um conjunto de processos. O objetivo dos *workflows* é fornecer um ambiente de programação especializado para minimizar o esforço de programação necessários para um cientista orquestrar um experimento científico apoiado por computador [20].

Atualmente, a ciência conta com as ferramentas de *workflows* como uma solução para automatizar as atividades envolvidas em seu ciclo de vida. Os experimentos científicos que envolvem a execução de um conjunto de programas encadeados podem ser modelados por meio de *workflows* científicos. Os *workflows* podem prover diversos benefícios no gerenciamento e automação de tarefas científicas. Yang *et al.* [103] apresenta 4 benefícios da adoção de *workflows*: (1) Facilidade, pois apoiam a fácil expressão do processo de execução e compartilhamento entre usuários; (2) Automação, pois diversas tarefas e recursos podem ser integrados em um *workflow* e executados sem interação/intervenção do usuário; (3) Eficiência, pois os motores de *workflows* podem permitir a execução eficiente automaticamente com base em determinadas metas e regras de otimização; e (4) Reprodutibilidade, pois as execuções dos *workflows* podem ser total ou parcialmente reproduzidas.

Um *workflow* pode ser modelado como um grafo direcionado em que os nós são transformações de dados implementadas por programas (*software*) e as arestas especificam as dependências dos fluxos de dados [50]. Os *workflows* incluem em geral 3 componentes: os processos (ou atividades), as dependências de dados e as artefatos (ou entidades). A Figura A.1 apresenta um modelo de *workflow* com 06 atividades $A = \{a_1, ..., a_6\}$, representadas com elipses, e as setas, que definem o fluxo de dados entre as atividades. Um *workflow* executa suas atividades orientadas por parâmetros de configurações e dados de entrada. Os estados e comportamento da sua execução são armazenados em uma base de dados, denominada base de proveniência.

A.1.1 Sistemas de Gerência de *Workflow* Científico

Os Sistemas de Gerência de Workflow Científicos (SGWfC), também denominados *Scientific Workflow Management Systems* (SWfMS), surgiram devido à queda da popularidade dos *scripts ad-hoc* para fins de representatividade de *workflows* científicos. Um SGWfC define, cria e gerencia a execução de *workflows* através do uso de *software*, capazes de interpretar a definição de processos, interagir com os participantes do *workflow* e, quando necessário, invocar o uso de ferramentas de TI

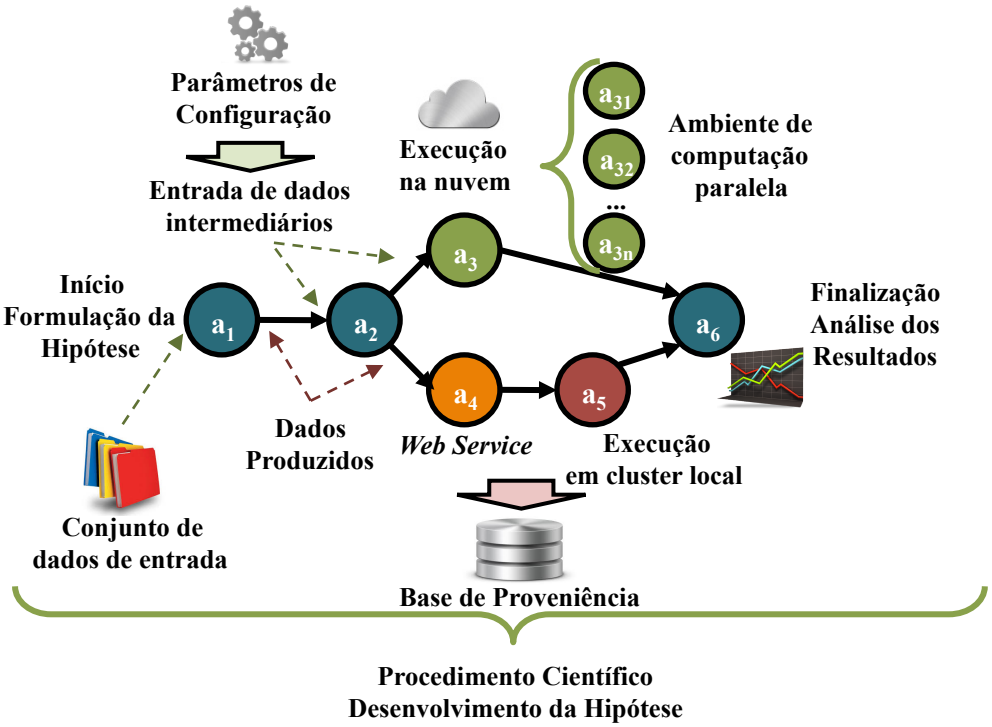


Figura A.1: Exemplo de *workflow* sob diferentes infraestruturas de computação.

(Tecnologia da Informação) e aplicações.

Os SGWfC fornecem uma forma simples e abstrata, através de interfaces gráficas, para modelagem e gerenciamento de *software* sem exigir que o pesquisador tenha conhecimentos avançados em programação [104]. Os *workflows* científicos geralmente são construídos pelos cientistas, no entanto, como não são especialistas em TIC, *software* ou redes de computadores, onde esses *workflows* operam, surge a necessidade de uma interface com o usuário robusta, tanto para a fase de construção do *workflow*, como para a fase de execução [82], essa interface é o SGWfC.

SciCumulus

O SciCumulus é um *middleware* leve concebido para distribuir, controlar e monitorar execuções paralelas de atividades de *workflows* científicos (ou mesmo *workflows* inteiros) a partir de um SWfMS em um ambiente de nuvem, como a Amazon EC2. O SciCumulus orquestra a execução de atividades de um *workflow* em um conjunto distribuído de máquinas virtualizadas [32]. Um dos principais objetivos do SciCumulus é obter as especificações do *workflow* e fornecer o paralelismo de dados automaticamente com suporte a execução de consulta de proveniência. Os dados são fragmentados a partir da varredura de um conjunto de combinações de parâmetros ou conjunto de dados de entrada. O processamento paralelo é obtido com *MapReduce* (Hadoop), porém, o mecanismo do SciCumulus é suportado por uma álgebra de *workflow*, permitindo a otimização e a programação dinâmica.

Vistrails

O Vistrails é um SGWfC que fornece suporte para simulações, exploração de dados e visualização. É um projeto código aberto implementado na linguagem *Python* que fornece uma infraestrutura que pode ser combinada com sistemas e bibliotecas existentes [105]. O Vistrails possui um repositório social para visualização chamado *crowdLabs*[1] que adota o modelo de redes sociais e integra um conjunto de ferramentas e uma infraestrutura escalável para fornecer um ambiente para analisar e visualizar dados de forma colaborativa. O repositório foi especificamente concebido para apoiar as necessidades computacionais dos cientistas, incluindo a capacidade de acessar computadores de alto desempenho e manipular grandes volumes de dados [106].

Taverna

O Taverna é um SGWfC de código aberto e de domínio independente com um conjunto de ferramentas usadas para projetar e executar *workflows* e ajudar em experimentos de simulações com modelos próximos aos reais. O Taverna foi criado pela equipe myGrid[2] [91]. O conjunto de ferramentas de *workflow* Taverna foi projetado para combinar *Web Services* distribuídos e/ou ferramentas locais em análises complexas. Essas execuções podem ser feitas em máquinas *desktop* locais ou através de uma maior infraestrutura, tais como supercomputadores, grades ou ambientes de nuvem. O SWfMS Taverna, possui o projeto *Wf4Ever*, que compromete-se a produzir uma arquitetura de *software* para a concepção e implementação da preservação de *workflow* com uma referência para a arquitetura e permitindo a preservação e recuperação eficiente de *workflow* em uma variedade de domínios[3].

A.2 Metadados e Dados de Proveniência

As colaborações científicas produzem conhecimento coletivo e recursos para resolver um problema particular ou para explorar uma determinada área de pesquisa. Esse contexto exige que os dados utilizados na geração de um produto tenham informações suficientes para que os demais cientistas possam interpretá-los e terem a confiança de utilizá-los em sua própria pesquisa científica [102]. Com isso, surge a necessidade de vincular o significado e a forma na qual esses dados foram produzidos usando dois conceitos importantes: os metadados e a proveniência.

Os metadados são dados descritores que atribuem um significado ao dado, ou seja, são dados sobre dados. Greenberrg[4] define metadados como dados que descre-

[1] http://www.crowdlabs.org/
[2] http://www.mygrid.org.uk
[3] http://www.wf4ever-project.org
[4] http://www.ils.unc.edu/janeg/

vem a estrutura de um objeto e as funções associadas a ele. Portanto, os metadados descrevem as características físicas e operacionais desse objeto. Já a proveniência é comumente definida como a origem ou histórico de derivação de algum objeto a partir da sua fonte original [78]. São as informações sobre os objetos e os processos envolvidos na produção de um conjunto de dados e informações, usados para formar avaliações sobre qualidade, confiabilidade ou confiança de um objeto [10]. A proveniência descreve as etapas nas quais os dados foram derivados e adiciona um valor significante a eles [80].

A proveniência pode ser prospectiva, captando a especificação de uma tarefa computacional, descrevendo os passos que precisam ser seguidos para a derivação de um resultado [52] ou, retrospectiva, que capta as etapas que foram executadas, bem como as informações sobre o ambiente de execução utilizado para obter dados específicos de um produto, criando um registro detalhado da execução de uma tarefa computacional. A proveniência é um elemento fundamental na derivação dos resultados obtidos com experimentos científicos, sem ela, os resultados de um *workflow* terão um valor limitado [107]. A comunidade científica tem adotado os modelos de referência OPM e o W3C PROV para a representação da proveniência. Ambos os modelos serão abordados nas seções A.2.1 e A.2.2.

A.2.1 *Open Provenance Model* (OPM)

O OPM (Modelo Aberto de Proveniência) é um modelo para representação de proveniência retrospectiva [108]. Ele foi desenvolvido durante as quatro edições do evento *Provenance Challenges* [109]. Os *Provenance Challenges* foram motivados pelos desafios e necessidades de entender as diversas formas de representações para proveniência, assim como os seus aspectos comuns e diferenças. Na segunda edição do desafio, a comunidade participante identificou a necessidade de conceber uma linguagem comum para representar os dados de proveniência, com o objetivo de facilitar a comunicação entre diversos sistemas. O resultado da segunda edição do desafio foi a concepção da primeira versão do OPM. No terceiro desafio, o modelo proposto anteriormente, foi avaliado, a fim de descobrir seus pontos falhos e futuras melhorias. No quarto desafio as mudanças foram efetivadas, originando a versão mais recente do modelo, a de revisão 1.1 [8].

O OPM foi projetado para atender o seguintes requisitos básicos [8]: (1) permitir que informações de proveniência possam ser trocadas entre sistemas, por meio de uma camada de compatibilidade com base em um modelo de proveniência compartilhado, (2) permitir que os desenvolvedores criem e compartilhem ferramentas que operam em tal modelo de proveniência, (3) definir a origem do dado de forma precisa, (4) apoiar a representação digital de proveniência para qualquer "coisa"produzida

por sistemas computacionais ou não, (5) permitir que vários níveis de descrição possam coexistir, e (6) definir um conjunto básico de regras que identifiquem as inferências válidas que podem ser feita na representação proveniência.

O OPM proporciona uma forma de representação de um conjunto de objetos de natureza qualquer, assim como os tipos de relação de dependência entre eles. O OPM representa esses objetos, e suas respectivas dependências, na forma de um grafo dirigido. Os nós modelam o conjunto de objetos e as ligações (*links*) modelam as relações de dependência. O conjunto de objetos representam os elementos digitais e não digitais (as "coisas") de um documento OPM. Esses objetos representam os resultados de simulações computacionais, objetos físicos, regras de decisão, estados e características de um objeto em um dado momento e assim por diante. Os objetos podem ser definidos em três categorias básicas do modelo:

- Artefato (A): representa um dado de estado imutável, que pode ter uma representação física em um corpo físico, ou digital, em um sistema de computador;
- Processo (P): representa ações realizadas ou causadas por artefatos, e resultam em novos artefatos;
- Agente (Ag): representa entidades contextuais agindo como um catalisador de um processo, permitindo, facilitando, controlando, ou afetando sua execução.

O OPM ajuda na captura das relações de dependências causais entre os elementos nó [8]. A Figura A.2 apresenta um grafo com as relações de dependência entre os agentes, artefatos e processos. As dependências identificam as ações executadas entre os objetos do OPM e são classificadas em cinco tipos: geração ($wasGeneratedBy(R)$), uso ($used(R)$), derivação ($wasDerivedFrom(R)$), controle ($wasControlledBy$) e gatilho de um processo para o outro ($wasTriggeredBy$). O relacionamento entre os objetos e as dependências são apresentados na Figura A.2. O sentido da seta caracteriza o relacionamento entre os objetos, origem representa a origem e o destino a causa. O significado de cada uma dessas dependências são listados abaixo [8]:

- A relação de uso (*used*) de um processo por um artefato é uma relação causal que indica que o processo requer a disponibilidade de um artefato para que ele possa completar a sua execução. Quando diversos artefatos são conectados em um mesmo processo por múltiplos arcos *used*, é determinada a necessidade da existência de todos eles para a conclusão de um processo.
- A relação de geração (*was generated by*) de um artefato para um processo é um relacionamento causal que significa que um artefato foi criado (gerado) por um determinado processo. Quando diversos artefatos são ligados a um mesmo

processo por múltiplos arcos *was generated by*, significa que um processo deve ter sido iniciado para que todos os artefatos tenham sido criados.

- A relação de ativação (*was triggered by*) do processo P_2 para um processo P_1 é uma dependência causal que indica que o início do processo P_1 foi requisitado pelo processo P_2. O processo P_1 somente inicia após o processo P_2 finalizar.
- A relação de derivação (*was derived from*) de um artefato A_2 para um artefato A_1 é um relacionamento causal que indica que o artefato A_1 precisa ter sido gerado para que A_2 também o seja. A criação de A_2 é dependente da existência de A_1.
- A relação de controle (*was controlled by*) de um processo P sob um agente Ag é uma dependência causal que indica que o processo P foi iniciado e finalizado sob o controle de Ag.

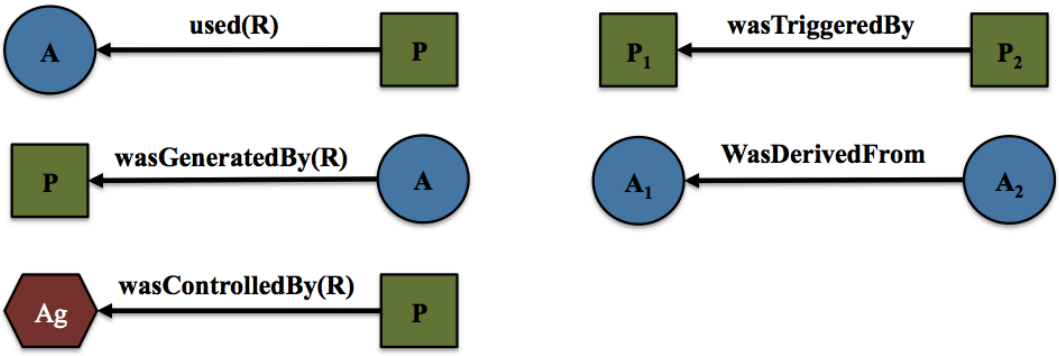


Figura A.2: Tipos de dependências do modelo OPM [8].

Algumas dependências incluem um texto opcional com a letra "R"entre parênteses. Ele indica a regra (*role*) que um agente exerce sob um processo. Essa regra é uma alternativa para representar as variadas funções entre artefatos, agentes e processos.

A.2.2 W3C Prov

O W3C PROV define um modelo de dados central para proveniência usado para a representação de entidades, pessoas e processos envolvidos na produção de um conjunto de dados ou objeto. Os objetivos são [9]: (1) permitir a troca de informação de proveniência entre diferentes sistemas, (2) permitir que desenvolvedores construam e compartilhem sistemas que operem sob tais modelos de proveniência, (3) definir proveniência de forma precisa e agnóstica, independente de tecnologia, (4) dar suporte a representação digital da proveniência para um artefato, produzido ou não

por sistema de computador, (5) permitir que múltiplos níveis de descrição coexistam, e (6) definir um conjunto de regras que identificam as inferências que podem ser feitas sob a representação da proveniência.

O W3C PROV é composto por uma família de 11 documentos cujo objetivo é permitir a ampla publicação e intercâmbio de proveniência na *internet* e em outros sistemas de informação [9]. Ele permite uma representação e a troca de informações de proveniência através de formatos, tais como XML, fornecendo definições para acessar informações de proveniência de forma a validá-las. A família de documentos da W3C PROV foi desenvolvida após a quarta edição do *Provenance Challenges* [109], e estão listados na Figura A.3. Os elementos apresentados na cor cinza foram utilizados no desenvolvimento do modelo de dados, restrições e para a criação dos documentos e base de proveniência desenvolvidos neste trabalho.

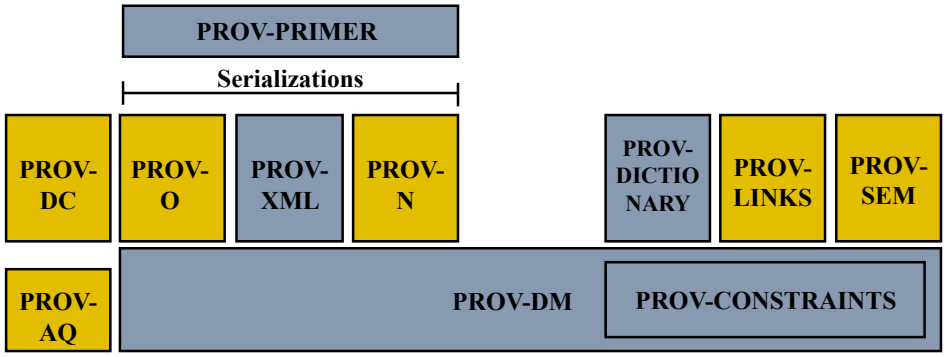


Figura A.3: Família de documentos W3C PROV. Figura adaptada de [9].

O elemento principal do W3C PROV é o seu modelo de dados (PROV-DM). Ele apresenta os componentes que descrevem a proveniência e as associações entre eles. Para manter os dados de proveniência válidos é utilizado o PROV-CONSTRAINT que apresenta todas as restrições para a construção do modelo de dados do Prov. O PROV-CONSTRAINT é usado para implementação de validadores de proveniência. O PROV-XML possui o esquema do modelo de dados do PROV e o PROV-PRIMER apresenta o primeiro modelo de dados do Prov.

Ele define um vocabulário comum para descrever a proveniência. Os tipo de dados e relações do PROV-DM são organizados de acordo com os seis componentes: (1) Entidades e atividades, (2) Derivações, (3) Agentes, responsabilidade e influência, (4) Bundles, (5) Alternação e (6) Coleção. A tabela A.1 apresenta os tipos e relações do W3C PROV. As letras R, L e T denotam as regras, localização e tempo.

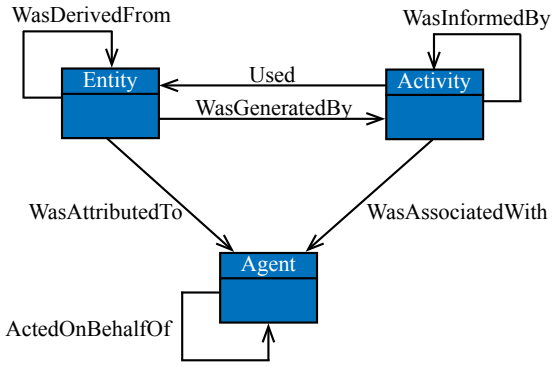


Figura A.4: Estruturas que formam o núcleo do Prov-DM [10].

Tabela A.1: Objetos e relacionamentos do W3C PROV.

		Objeto					
		Entity		**Activity**		**Agent**	
Assunto	**Entity**	WasDerivedFrom Revision Quotation PrimarySource AlternateOf SpecializationOf HadMember		WasGeneratedBy WasInvalidatedBy	*R* *T* *L*	WasAttributedTo	
	Activity	Used WasStartedBy WasEndedBy	*R* *T* *L*	WasInformedBy		WasAssociatedWith	*R*
	Agent	-		-		ActedOnBehalfOf	

Atividades e Entidades

Esses elementos se concentram nos tipos atividades e entidades e nas relações entre eles. As relações que compõem essa classe de componentes são formadas pelo uso, geração, inicialização, finalização, invalidação e informação.

Um tipo Entidade (*Entity*) é um conceito amplo, descrito como uma coisa física, digital ou conceitual que pode ser real ou abstrato. Pode ser representado através de um objeto digital ou físico [79]. Um tipo Atividade (*Activity*) é algo que ocorre ao longo de um período de tempo e atua sobre ou com entidades. Essa atuação pode incluir o consumo, processamento, transformação, modificação, realocação, uso, geração ou ser associado a entidades. Uma atividade representa um processo que executa uma determinada ação.

As atividades são associadas com as entidades por meio da relação de Geração (*Generation*) de novas entidades ou pelo Uso (*Used*) de uma entidade já existente. A geração é o processo onde uma atividade produz uma nova entidade. Essa nova entidade não existia até o momento da sua criação, e após ser criada, está apta a ser usada por qualquer uma das atividades pela relação de uso. As relações de geração e uso de entidade formam uma Comunicação (*Communication*) entre duas atividades, onde uma atividade produz uma entidade que será consumida por uma outra.

A Inicialização (*Start*) é o processo no qual uma entidade, denominada entidade

gatilho (*trigger entity*), dispara o início da execução de uma atividade. Uma vez que uma atividade tenha sido inicializada, a sua execução é instantânea. Após uma atividade ter sido inicializada e executada, ela passa pelo processo de Finalização (*End*). A finalização é disparada por uma entidade gatilho, fazendo com que a atividade deixe de existir instantaneamente.

A Invalidação (*Invalidation*) é a relação em que uma atividade sinaliza uma entidade como destruída, cessada ou expirada. A entidade continua a existir nos registros da proveniência, porém, torna-se indisponível para outras ações. Uma entidade possui uma duração que é iniciada com a geração dessa entidade por alguma atividade, portanto, a invalidação marca o fim da duração de um entidade.

Derivações

O componente de derivação trata a transformação de uma entidade em outra, uma atualização de uma entidade resultando em uma nova, ou a construção de uma nova entidade com base em uma entidade pré-existente [9]. A derivação é formada por um conjunto de subtipos denominados Revisão ($WasRevisionOf$), Citação ($WasQuotedFrom$) e Fonte Primária ($HasPrimarySource$).

Uma Derivação (*Derivation*) é a transformação de uma entidade em outra. Ela é o resultado da transformação, criação, atualização de uma entidade por uma atividade. A Revisão (*Revision*) é uma derivação em que uma entidade resultante é uma versão revisada de uma original. A entidade revisada possui um conteúdo substancial a partir de uma entidade original. Uma Citação (*Quotation*) é o processo de referenciar uma entidade. É aplicado quando uma entidade é utilizada por alguém que não é o autor original e determina que uma entidade foi copiada ou citada.

Uma Fonte Primária (*Primary Source*) identifica alguma coisa produzida por um agente com experiência direta e conhecimento sobre o tema. É uma espécie de relação de derivação de materiais secundários através de suas fontes primárias. É importante observar que uma entidade pode ser fonte primária para uma determinada entidade, mas não para outra [9].

Agentes, Responsabilidade e Influência

O agente é definido como algo que tem uma responsabilidade pela execução de uma atividade, existência de uma entidade ou pela atividade de um outro agente. Nas definições do W3C PROV-DM [9], é útil definir algumas categorias básicas de agentes a partir de uma perspectiva de interoperabilidade. Por tal motivo, são definidos 3 tipos básicos de agentes: agente de *software* (*software agent*), pessoa (*person*) e organização (*organization*).

Um agente está relacionado com três tipos de associações binárias, denominadas

associação, atribuição e delegação. Na Atribuição (*Attribution*), uma entidade é atribuída a um agente. Uma atribuição é útil quando uma entidade foi gerada por uma atividade que não foi especificada, ou pelo fato de não ser conhecida ou por ser irrelevante. Uma Associação (*Association*) é a assimilação da responsabilidade de um agente a uma entidade, indicando que o agente desempenha um papel na atividade [9]. A associação pode conter um tipo denominado plano *plan*, que é uma entidade que representa um conjunto de ações ou passos destinados a um ou mais agentes para a realização de certos objetivos.

Uma Delegação (Delegation) é a atribuição de autoridade e responsabilidade de um agente para a realização de uma atividade específica como um delegado ou representante. Um agente delega a responsabilidade de realização de uma atividade para outro agente.

Bundles

Um *Bundle* é um mecanismo de suporte de proveniência sobre a proveniência. É um mecanismo de confiança que permite que um documento de proveniência tenha informações de proveniência. Ele registra informações sobre o momento em que o documento foi criado e como a proveniência foi atribuída.

Entidades Alternativas

Esse componente envolve o tipo de dado entidade e as duas relações binárias reflexivas denominadas especialização e alternação. Uma Especialização (*Specialization*) é uma entidade que possui todos os aspectos de uma entidade mais genérica, e possui ainda, aspectos mais específicos em relação as demais entidades. Duas entidades Alternativas, ou Alternadas (*Alternate*, apresentam aspectos de uma mesma coisa. Esses aspectos podem ser iguais ou diferentes, e as entidades alternadas podem ou não sobrepor-se no tempo [9].

Coleções

Trata-se de uma entidade que fornece uma estrutura para alguns componentes que devem ser entidades. Cada um dos componentes será um membro das coleções. Existem muitos tipos de coleções, tais como: conjuntos, dicionários ou listas.

A.3 A Computação em Nuvem

A computação em nuvem, segundo a definição do *National Institute of Standards and Technology* (NIST) [25], é um modelo que permite o acesso onipresente, conveniente e sob demanda a um conjunto de recursos computacionais configuráveis através de

uma rede de computadores, permitindo que tais recursos possam ser rapidamente fornecidos e liberados com o mínimo esforço de gerenciamento e interação com o fornecedor do serviço. Este modelo de computação foi concebido para fornecer recursos de forma flexível, altamente escalável e sob demanda.

A computação em nuvem é fundamentada em quatro tecnologias: grade computacional, computação de utilidade, virtualização de recursos e arquitetura orientada a serviços. A grade computacional é caracterizada como uma forma de computação distribuída que permite que diversos recursos de computadores heterogêneos conectados a rede de computadores trabalhem no processamento de uma tarefa complexa ao mesmo tempo. A ideia da computação em nuvem envolve os conceitos de grande computacional combinados com a escalabilidade e disponibilidade [27].

A computação de utilidade é um modelo de fornecimento de serviços em que um fornecedor disponibiliza os recursos de computação na forma de um serviço. O fornecimento de recursos adota alguma métrica clara, tal como quantidade de recursos adquiridos, perfil dos recursos, período de utilização, serviço embarcados (*backup*, alta disponibilidade e etc.) para orientar a aquisição de um serviço. Na computação de utilidade é necessário determinar diferentes parâmetros de qualidade de serviço (*Quality of Service - QoS*) para garantir que os requisitos do usuário sejam satisfeitos. Para isso é adotado um modelo de utilidade para balancear o fornecimento e a demanda de recursos por meio de um contrato formal para a garantia da qualidade do fornecimento de serviços segundo uma SLA [28].

A virtualização de um ambiente de computação permite que vários recursos de *hardware* e *software* sejam vistos e gerenciados na forma de um *pool* de recursos. O principal objetivo é centralizar o gerenciamento, otimizar o uso dos recursos de computação e usar a capacidade disponível da forma mais eficiente possível entre os usuários e as aplicações. A virtualização cria uma camada de abstração de recursos físicos, tais como computador, armazenamento e rede para a forma de recursos lógicos. O principal objetivo é fazer com que um computador físico possa acomodar diversos computadores lógicos, denominados máquinas virtuais [29]

A arquitetura orientada a serviços (*Service Oriented Architecture* - SOA) é uma abordagem que utiliza serviços disponíveis na rede, com o objetivo de oferecer um conjunto específico de funções. Os recursos de *software* em SOA são empacotados como serviços bem definidos que fornecem um conjunto de funcionalidades padronizadas independentes de outros serviços [30].

A.3.1 Características Essenciais da Nuvem

A nuvem possui 5 características essenciais: rápida elasticidade, serviço de medição, agrupamento de recursos para compartilhamento (*pool* de recursos), auto atendi-

mento sob demanda e amplo acesso a rede [25]. O auto atendimento sob demanda permite obter recursos de forma automática, sem intervenção humana, por meio de uma interface que apresente um catálogo de recursos e os serviços relacionados. Os provedores de nuvem disponibilizam uma interface que permite a apresentação e a requisição de tais recursos. O amplo acesso a rede de computadores permite que os recursos da nuvem possam ser acessados remotamente, a partir de diferentes plataformas operacionais, tais como estações de trabalhos, dispositivos móveis e demais dispositivos caracterizados como cliente magro (*client thin*). Este acesso geralmente ocorre por meio da *internet*, permitindo que os recursos da nuvem possam ser amplamente acessado a partir de qualquer local que tenha suporte a *internet*.

O agrupamento de recursos para compartilhamento permite encontrar um grande número de opções de recursos de forma flexível para que o cliente possa selecionar os que precisa, maximizando o atendimento aos requisitos do negócio e minimizando os custos financeiros para a aquisição dos recursos. Os recursos são formados em geral pelos componentes de computação (CPU e memória), armazenamento em massa (discos) e rede de computadores. Esse conjunto de recursos são dinamicamente alocados para múltiplos usuários baseado em um modelo de multi alocações (*multi-tenant*) [26]. O compartilhamento de recursos é possível por meio de técnicas de virtualização de *hardware* (máquinas virtuais, disco e redes locais virtualizadas).

A elasticidade é a habilidade de se expandir e reduzir a alocação dos recursos de TI de forma eficiente, ou seja, pode escalar para obtenção de mais recursos (*scale up*) ou escalar para redução de recursos (*scale down*) dinamicamente. Pode-se alocar inicialmente um número mínimo de recursos e posteriormente expandi-lo baseado em sua necessidade, dando a impressão que a nuvem pode prover o acesso a infinitos recursos de TI. A alocação e liberação dos recursos pode ser feita de maneira manual pelo próprio cliente ou de forma automática mediante critérios e restrições previamente definidas e explicitadas por meio da SLA.

A medição do serviço de fornecimento de recursos TI são cobrados baseado na medição de uso dos recursos de TI. O serviço de medição utiliza diversos monitores de uso de recursos geralmeente baseados na quantidade de recursos alocado, tais como o tempo de CPU, largura de banda, capacidade de armazenamento, em relação a quantidade de tempo em que o recurso foi usado. O modelo de fornecimento e de medição caracterizam o custo operacional na forma *"pay-as-you-go"*, ou seja, o cliente paga pelos recursos alocado baseado na forma como ele utiliza, tornando o processo transparente para ambas as partes, cliente e provedor.

A.3.2 Modelos de Serviço de Nuvem

Um fornecedor de nuvem pode prover 3 tipos de modelo de serviços aos usuários [27] [110]: (1) *Software* como um serviço (*Software as a Serviçe* – SaaS), (2) Plataforma como um serviço (*Plataform as a Service* – PaaS) e (3) Infraestrutura como um serviço (*Infrastructure as a Service* – IaaS). O modelo de serviço IaaS ocorre em nível de recursos de computação, armazenamento e redes de computadores. Trata-se da camada base da pilha de fornecimento de recursos da nuvem. Esse tipo de fornecimento é possível devido a tecnologia de virtualização que permite dividir e alocar os recursos dinamicamente baseado na demanda. O modelo IaaS permite aos usuários implantar e executar suas aplicações e sistema operacional em VMs.

No modelo de serviço PaaS, os fornecedores de nuvem podem oferecer um nível de abstração adicional, em vez de fornecer uma infraestrutura virtualizada, eles podem fornecer a plataforma de *software* onde os sistemas possam executar [27]. Com isso, o dimensionamento dos recursos de *hardware* exigidos pela execução dos serviços é feita de forma transparente. No modelo de serviço SaaS, os aplicativos e *softwares* são hospedados remotamente, permitindo que os usuários acessem e manipulem os aplicativos via *internet*. Esse modelo elimina a necessidade de instalação e configuração do *software* no computador do usuário.

Alguns autores como Wang *et al.* incluem um quarto modelo de serviço denominado de Dados como um Serviço (*Data as a Service* - DaaS). Nesse modelo, os dados de vários formatos e múltiplas origens podem ser acessados por meio de serviços fornecidos na rede de computadores, para que os usuários possam manipulá-los remotamente como se estivessem com os dados em seu disco local [111].

A.3.3 Reprodução com a Infraestrutura da Nuvem

A computação em nuvem tem o potencial de revolucionar a *e-Science* dando aos cientistas os recursos computacionais necessários no momento em que eles precisam [100]. Atualmente, os governos, institutos de pesquisa e líderes das indústrias estão adotando a computação em nuvem com o propósito de resolver os seus crescentes problemas de demanda de computação e armazenamento de dados [110]. Como a nuvem fornece recursos de computação de forma escalável e elástica, ela torna-se uma solução atraente para os experimentos de pesquisa na *e-Science* [23].

A computação em nuvem pode trazer diversas vantagens nos mais variados cenários para a comunidade científica. Segundo Armbust [24], em um primeiro cenário, uma organização pode optar por pagar apenas pelas horas que usa os recursos. Se uma organização usa um *datacenter* apenas 08 horas/dia, pode não ser vantajoso adquirir e manter uma infraestrutura para funcionar por apenas 1/3 do período útil. Os gastos com essa infraestrutura de médio a longo prazo pode ser

maior do que adquirir as 08 horas diárias em um fornecedor de nuvem.

Em um segundo cenário, quando a demanda é desconhecida pode ocorrer picos de demanda de recursos, seguido de uma redução, ou seja, uma oscilação na utilização dos recursos. Esta característica pode ser amenizada com os conceitos de elasticidade proposto pelo paradigma de computação em nuvem. Com a elasticidade os recursos são alocados/liberados baseado na demanda da atividade. Já em um terceiro cenário, os ambientes que executam processamentos em lotes se beneficiam com esse paradigma, pois, o processamento pode ser feito de uma forma mais rápida alocando-se de uma vez a quantidade de recursos necessários para executar a tarefa em um período de tempo menor. Por exemplo, usar 1.000 máquinas/hora tem o mesmo custo de utilizar 01 máquina por 1.000 horas [24].

Yang *et al.* [103] destacam que a nuvem pode ser usada na *e-Science* para fornecer infraestrutura de TI escalável, serviços QoS confiáveis e um ambiente de computação personalizável. Os cientistas recebem um ambiente com recursos abundantes e elasticidade tanto para escalar o uso dos recursos quanto para liberá-los quando não são necessários. Em geral, os objetivos envolverm minimizar o tempo total de conclusão de um processamento com uma restrição orçamentária ou minimizar o processamento sob uma restrição de tempo. A nuvem fornece oportunidade para os pesquisadores testarem suas ideias e códigos antes de investirem recursos financeiros mais significativos em infraestrutura em uma escala potencialmente maior [15].

A computação em nuvem pode auxiliar a reprodução de diversas maneiras [38]: (1) na camada de dados a nuvem permite que os conjuntos de dados possam ser facilmente armazenados e compartilhados virtualmente, sem necessariamente copiá-lo para outro computador. A *Amazon Web Services* (AWS) já tomou a iniciativa de arquivar muitos conjuntos de dados públicos no catálogo *on-line* [5]. Ele inclui dados das áreas de astronomia, biologia, química e climatologia. (2) na camada do sistema, *snapshots* de sistemas de computador completos para eles possam ser trocados entre cientistas. Cópias de computadores armazenados e compartilhados em uma VM na nuvem podem fazer as análises computacionais mais reprodutíveis. (3) a camada de serviço, os serviços computacionais são expostos a aplicações externas por meio de alguma forma de programação de interface, tais como o *Entrez Utilities* da NCBI [6], que fornece acesso aos dados e recursos computacionais do seu repertório.

Alguns autores discutem os benefícios e as vantagens de migrar o ambiente de testes experimental das estações de trabalhos, *clusters* e grades para o ambiente de nuvem computacional. As características e os recursos do provedores de nuvem podem aumentar a capacidade de reprodutibilidade científica. Howe [89] apresenta

[5] http://aws.amazon.com/publicdatasets/

[6] http://eutils.ncbi.nlm.nih.gov

um conjunto de características da nuvem que ajudam a justificar essa afirmativa:

- Maior captura de variáveis: é possível compartilhar todo o ambiente de trabalho (dados, códigos, *logs*, históricos de uso, resultados intermediários, figuras, anotações, experimentos fracassados, detalhes de configuração do sistema operacional e demais informações) por meio de VMs.

- Portabilidade: a VM torna-se a unidade de compartilhamento da pesquisa. Esse nível de abstração elimina grande parte dos problemas de portabilidade nos processos de instalação, configuração e execução de *softwares* de terceiros.

- Menos restrições sobre o métodos de pesquisa: experimentos executados em um computador local também podem ser executados em uma VM na nuvem, exceto em casos com *hardware* especializado. Em geral, não é necessário alterar a metodologia de trabalho para usar uma VM, pois é possível usar outros sistemas operacionais, linguagens, bibliotecas, ferramentas, convenções e práticas tendo apenas o custo de estabelecer um ambiente virtual no qual irá trabalhar.

- *Backups*: *snapshots* das VMs armazenados em intervalos regulares fornecem um diário da ações e processos no laboratório. Para obter um estado anterior do processo basta identificar a VM do período desejado e instanciá-la.

- VMs como publicações: VMs hospedadas nos provedores de nuvem são associadas a um identificador único e podem ser referenciadas por artigos científicos.

- Código, dados, ambiente e recursos: uma VM empacota os códigos, dados e ambiente operacional devidamente instalados e configurados. Além disso, a infraestrutura da nuvem fornece um ambiente de computação para a reprodução equivalente ao que foi usado para a execução do experimento. Além disso, é possível fazer o *download* da VM para o ambiente de trabalho local.

- Queda no preço: o preço das unidades de computação e armazenamento tem diminuído na maioria dos cenários de ofertas dos provedores de nuvem, em muitos casos devido a queda do preço do *hardware*.

- Reprodutibilidade de arquiteturas complexas: experimentos computacionais estão usando cada vez mais arquiteturas complexas, tais como, servidores de banco de dados e arquivos, técnicas de computação de alto desempenho, programação para unidades de processamento gráfico e etc. A nuvem vem oferecendo recursos e ferramentas para experimentos aplicado a esses cenários.

- Experimentos colaborativos sem restrição: uma instância de VM compartilhada na nuvem permite que dois ou mais pesquisadores possam trabalhar colaborativamente em um experimento, cada um utilizando a sua conta.

- Compartilhamento de grandes conjuntos de dados: a ciência têm se tornado intensiva de dados nos mais variados campos do conhecimento. Nesse cenário, a obtenção ou troca de grandes conjuntos de dados de uma estação de trabalho local para outra pode ser inviabilizada devido ao volume desses dados chegar a escalas muito grande. Portanto, é mais interessante hospedar os dados nos provedores de nuvem e migrar a computação para o local onde estão os dados.

- Compartilhamento de custos: a reprodução envolve questões de custos, onde os autores arcam com os custos de armazenamento da VM e os experimentadores e aqueles que desejam reproduzir arcam com os custos de reexecução.

- Compatibilidade com outras abordagens: a definição dos experimentos encapsulados em VMs é compatível com outras abordagens de reprodução. Se os pesquisadores padronizarem as linguagens de programação SGWfC, formato de arquivos de metadados essas técnicas podem ser usadas em uma VM da mesma forma que poderiam ser usadas em um computador físico.

As características do paradigma de nuvem permitem atender a premissa de que para fazer a reprodução dos experimentos é necessário ter acesso a toda infraestrutura de *hardware* e *software* configurados de forma equivalente ao experimento original, bem como ter acesso ao conjunto de dados originalmente utilizados [23].

A.4 Considerações Finais

A captura e o armazenamento dos metadados e proveniência ajudam a rastrear características importantes necessárias para a reprodução. No entanto, é necessário avaliar alguns pontos importantes como a quantidade de informação sobre metadados e proveniência, que podem ser maior do que os dados que ela está descrevendo. É necessário atentar para questões de granularidade de uma informação, bem como, avaliar a possibilidade de representação da informação em um conjunto de camadas. Os dados de proveniência devem ser imutáveis, ou seja, uma vez que um dado ou um evento de derivação de dados foi registrado, ele não pode ser atualizado ou removido. Qualquer modificação no estado desses dados deve ser inserido como um complemento ou uma nova versão no mecanismo de armazenamento. O compartilhamento dos experimentos em um provedor de nuvem pública permite que os objetos de pesquisa vinculados ao experimento fiquem acessíveis a outros usuários que desejam utilizá-lo. Além disso, com o avanço no compartilhamento de grandes bases de dados científicas em provedores de nuvem incentivam a criação da infraestrutura do laboratório utilizando a infraestrutura do provedor que está hospedando essa base de dados.