Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Gerenciamento de Proveniência de Dados de Workflows de Bioinformática em Ambiente de Nuvem Computacional

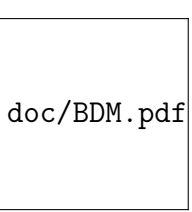
Fernanda Hondo Tedesque

Dissertação apresentada como requisito parcial para
conclusão do Mestrado em Informática

Orientadora
Prof.ª Dr.ª Maristela Terto de Holanda

Coorientadora
Prof.ª Dr.ª Aletéia Patrícia Favacho de Araújo

Brasília
2018


doc/BDM.pdf

Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Gerenciamento de Proveniência de Dados de Workflows de Bioinformática em Ambiente de Nuvem Computacional

Fernanda Hondo Tedesque

Dissertação apresentada como requisito parcial para
conclusão do Mestrado em Informática

Prof.ª Dr.ª Maristela Terto de Holanda (Orientadora)
CIC/UnB

Prof.ª Dr.ª Célia Ghedini Ralha
CIC/UnB

Prof. Dr. Sergio Lifschitz
PUC/Rio

Prof. Dr. Bruno Luiggi Macchiavello Espinoza
Coordenador do Programa de Pós-graduação em Informática

Brasília, 25 de Junho de 2018

Dedicatória

Dedico este trabalho aos meus amados pais. Dedico também aos meus irmãos e à minha família. Dedico ao Vinícius e à Polyane, pelo suporte, pelo apoio e pela força.

Agradecimentos

Agradeço, primeiramente e acima de tudo, à Deus, pois mesmo em meus dias mais sombrios e de pouca fé, sempre esteve ao meu lado, me levantou e me fez continuar.

Agradeço aos meus pais que, embora distantes durante este trabalho, são a minha maior motivação para lutar. Agradeço à minha mãe pelos ensinamentos, pela confiança e todo suporte necessário para concluir este trabalho. Ao meu pai, pelo carinho e conforto oferecidos em cada ligação telefônica recebida.

Agradeço aos meus irmãos e companheiros de vida, Rafael e Daniel, por terem me proporcionado toda amizade, todo amor e por terem me permitido amar e cuidar dos melhores presentes que já recebi: meus sobrinhos, Isabele, Theo, Lucca e Maitê.

Também agradeço ao Vinícius, pelos anos de companheirismo vividos, por sempre acreditar na minha capacidade, no meu potencial e na minha força.

Agradeço à minha amiga Polyane, pelo acolhimento, pelo apoio, pelo suporte, pelas histórias, risadas e todas as desventuras em série vividas e compartilhadas.

Agradeço também às melhores amigas "da minha terra", pelo apreço e pelo apoio sempre que precisei que uma palavra amiga. E por serem sempre presentes em minha vida, mesmo que distantes.

Agradeço às minhas orientadoras: Maristela e Aletéia. À professora Maristela, pela sabedoria, pelos conselhos, indicações e por me permitir viver toda essa experiência. À professora Aletéia, pela atenção e orientações que conduziram o estudo e escrita desta dissertação.

Agradeço a todos amigos que fiz aqui em Brasília, que se tornaram minha família e que me ajudaram a sempre ser perseverante.

Agradeço a todos que participaram direta e indiretamente deste caminho e que contribuiram de alguma forma para que este trabalho pudesse chegar até aqui.

Agradeço à CAPES pela concessão de bolsa de mestrado.

Resumo

Os experimentos da biologia molecular são frequentemente apresentados sob a forma de *workflows* científicos. Um *workflow* científico é composto por um conjunto de atividades realizadas por diferentes entidades de processamento através de tarefas gerenciadas. O conhecimento sobre a trajetória dos dados ao longo de um determinado *workflow* permite a reprodutibilidade por meio da proveniência de dados. Para reproduzir um experimento de Bioinformática *in silico*, é preciso considerar outros aspectos, além das tarefas executadas em um *workflow*. De fato, as configurações computacionais nas quais os programas envolvidos são executados são um requisito para a reprodutibilidade. A tecnologia da computação em nuvem pode ocultar detalhes técnicos e facilitar ao usuário a configuração desse ambiente sob demanda. Os sistemas de banco de dados NoSQL também ganharam popularidade, particularmente na nuvem. Considerando este cenário, é proposta uma modelagem para a proveniência de dados de experimentos científicos, em ambiente de nuvem computacional, utilizando o PROV-DM e realizando o mapeamento para três diferentes tipos de famílias de sistemas de banco de dados NoSQL. Foram executados dois *workflows* de Bioinformática envolvendo diferentes fases, os quais foram utilizados para os testes nos bancos de dados NoSQL Cassandra, MongoDB e OrientDB, e em seguida é apresentada uma análise dessas execuções e testes. Os resultados obtidos mostraram que os tempos de armazenamento da proveniência são mínimos comparados aos tempos de execução dos *workflows* sem o uso da proveniência e, portanto, os modelos propostos para os bancos de dados NoSQL mostraram ser uma boa opção para armazenamento e gerenciamento de proveniência de dados biológicos.

Palavras-chave: Banco de Dados, Computação em Nuvem, Bioinformática

Abstract

Molecular biology experiments are often presented in the form of scientific workflows. There is a set of activities performed by different processing entities through managed tasks. Knowledge about the data trajectory throughout a given workflow enables reproducibility by data provenance. In order to reproduce an *in silico* bioinformatics experiment one must consider other aspects besides those steps followed by a workflow. Indeed, the computational settings in which the involved programs run is a requirement for reproducibility. Cloud computing technology may hide the technical details and make it easier for the user to set up such an on-demand environment. NoSQL database systems have also gained popularity, particularly in the cloud. Considering this scenario, a model for the provenance of data from scientific experiments in a computational cloud environment is proposed, using the PROV-DM and mapping to three different types of families of NoSQL database systems. Two Bioinformatics workflows involving different phases were performed, which were used for the tests in the NoSQL Cassandra, MongoDB and OrientDB databases, followed by an analysis of these executions and tests.The results obtained showed that the storage times of the provenance are minimal compared to the execution times of the workflows without the use of the provenance and therefore, the proposed models for the NoSQL databases proved to be a good option for storage and management of biological data.

Keywords: Data Base, Cloud Computing , Bioinformatics

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

A Bioinformática é uma área multidisciplinar que une conhecimentos da matemática, estatística, ciência da computação e biologia molecular, e surgiu da necessidade de utilização de recursos computacionais eficientes para armazenar a grande quantidade de dados adivindos de projetos da biologia, como sequências de DNA (*deoxyribonucleic acid*) e RNA (*ribonucleic acid*) produzidas pelos sequenciadores automáticos de DNA, além da utilização de plataformas computacionais eficientes para analisar e interpretar os resultados obtidos (PROSDOCIMI *et al.*, 2002).

Os projetos da Bioinfomática geralmente são modelados como *workflows*. Define-se *workflow* a automação de processos em que tarefas são passadas entre participantes de acordo com um conjunto de regras para atingir um objetivo (HOLLINGSWORTH, 1995). Os *workflows* são compostos por diferentes fases ou atividades dispostas em ordem de execução, por exemplo, uma pesquisa de uma base de dados de proteínas, seguida por uma análise de alinhamento, e por um filtro definido pelo usuário (GOBLE, 2002). Assim, projetos de Bioinformática utilizam diferentes ferramentas computacionais cujas configurações e parâmetros de entrada podem afetar fortemente os resultados obtidos (DE PAULA, 2012).

Os *workflows* científicos podem ser beneficiados em qualidade e confiabilidade quando as informações de sua execução são coletadas e armazenadas para análises e possíveis reexecuções. Para que pesquisadores detenham maior controle, maior confiabilidade e possam realizar análises minuciosas sobre seus experimentos científicos, a proveniência de dados tem grande importância. A proveniência de dados é utilizada em diversas áreas da ciência, e em áreas multidisciplinares como a Bioinformática. De acordo com Buneman *et al.* (2001), em tradução livre "a procedência do dado é a descrição da origem de um pedaço de dado e o processo pelo qual este chegou em um banco de dados". Dessa forma, a proveniência de dados tem se mostrado um requisito de suporte muito importante aos pesquisadores para a administração das execuções de seus experimentos. Por exemplo, a

partir de informações de proveniência é possível verificar a qualidade dos dados gerados, porque pode-se analisar seus dados ancestrais e determinar se eles são confiáveis ou não (MARINHO *et al.*, 2009). De acordo com Paulino *et al.* (2010), sem o dados de proveniência, o experimento tem sua avaliação e reprodução comprometidas e é fundamental que os cientistas saibam quais os parâmetros foram utilizados e quais os produtos de dados foram gerados.

A coleta e o gerenciamento de proveniência de *workflows* executados em ambientes distribuídos são tarefas complexas. *Workflows* científicos, quando executados em ambientes distribuídos, como nuvens computacionais, aumentam consideravelmente o volume de recursos e ferramentas utilizadas em sua execução, aumentando, consequentemente, a quantidade de dados de proveniência a serem gerenciados. Há vários cenários de execução de *workflows* em ambiente distribuídos, cada um com características que dificultam o gerenciamento da proveniência (MARINHO *et al.*, 2009). Em cada cenário é importante que se conheça a melhor estratégia de manipular, armazenar e recuperar os dados de proveniência.

Quando executados em cenário de nuvens computacionais, os *workflows* científicos obtém diversas vantagens, principalmente quanto à elasticidade de recursos. Assim que o cientista verifica a necessidade de mais recursos para execução de seus experimentos, basta que o mesmo faça uma solicitação ao provedor da nuvem e os recursos são disponibilizados. Desta forma, a comunidade científica têm migrado seus experimentos para ambientes de nuvem, tornando cada vez mais necessária a coleta de dados de proveniência na nuvem, pois é preciso assegurar a reprodutibilidade desses experimentos. Uma das vantagens da nuvem para os experimentos é prover aos cientistas o acesso a uma grande variedade de recursos sem ter que necessariamente adquirir e configurar a infraestrutura computacional (PAULINO *et al.*, 2010).

Gerenciar a quantidade de dados de insumos dos *workflows* científicos torna-se um desafio já que essa quantidade tende a aumentar significativamente, demandando das bases de dados maior escalabilidade. Muitos sistemas gerenciadores de experimentos científicos utilizam bancos de dados relacionais para armazenar dados de proveniência por serem mais estáveis e consolidados. Soluções alternativas para esse desafio estão sendo propostas, como a utilização de bancos de dados não relacionais, também conhecidos como NoSQL (*Not ONly SQL*).

Neste contexto, é proposto o desenvolvimento de uma modelagem de dados de proveniência de execuções de *workflows* de Bioinformática em nuvens computacionais, com foco no armazenamento e processamento desses dados. A modelagem deve ser capaz de refletir as informações sobre execuções do *workflow* como, arquivos utilizados pelo *workflow*, arquivos gerados nas fases do experimento, atividades executadas, máquinas utilizadas,

ambiente de execução dentre outras informações. Usando esse modelo, os biólogos e pesquisadores podem acessar detalhes de uma execução específica de um fluxo de trabalho, comparar resultados produzidos por diferentes execuções e planejar novos experimentos com mais eficiência.

Para avaliar a modelagem proposta por este trabalho, foi feita a análise da modelagem quanto ao gerenciamento dos dados de proveniência com foco em seu armazenamento, recuperação e geração dos grafos de proveniência em diferentes sistemas de gerenciamento de banco de dados NoSQL.

1.1 Descrição do Problema

Workflows científicos de projetos da Bioinformática, em ambientes distribuídos como nuvens computacionais, podem ser executados diversas vezes, com diferentes parâmetros de entrada, por diferentes equipes, em localizações geográficas distintas e podem produzir diferentes dados como resultado. Tanto os dados produzidos, nas fases do *workflow*, quanto sua proveniência necessitam ser armazenados com eficiência para que futuras execuções e análises sobre esse mesmo experimento possam ser feitas.

Conforme o cenário apresentado, pode-se citar como questão de pesquisa: é possível definir uma modelagem para dados de proveniência de execução de experimentos científicos em nuvem, que possibilite o gerenciamento e análise simples de experimentos científicos em banco de dados NoSQL?.

Neste sentido, este trabalho assume como hipótese que é possível definir as principais informações sobre a execução dos *workflows* da Bioinformática em nuvem e compor uma modelagem baseada no PROV-DM para gerenciamento e análise de experimentos.

1.2 Objetivos

O objetivo geral deste trabalho é o desenvolvimento de uma modelagem para o gerenciamento da proveniência de dados de experimentos científicos da Bioinformática modelados como *workflows* científicos e executados em ambientes de nuvens computacionais. Para que o objetivo geral seja atingido, foram definidos os seguintes objetivos específicos:

- Definição da modelagem de dados de proveniência de *workflows* da Bioinformática baseada no PROV-DM;
- Definição do modo de integração entre as ferramentas para a coleta, armazenamento e recuperação dos dados de proveniência;

- Execução *workflows* científicos reais da Bioinformática para validação das modelagens de dados para os NoSQL;
- Avaliação dos resultados obtidos.

1.3 Metodologia e Estrutura do Trabalho

Este trabalho de mestrado tem natureza empírica com propósito de pesquisa exploratória e abordagem qualitativa. Foram realizadas pesquisas bibliográficas e realizados dois estudos de caso. Uma investigação sobre *workflows* científicos, composição, estrutura e áreas de aplicação, foi realizada para que se pudesse definir os principais elementos e informações que os compõem. O foco deste trabalho foi voltado para *workflows* sequenciais utilizados na Bioinfomática. Um estudo sobre o modelo PROV-DM também foi realizado, para que os elementos de proveniência pudessem ser moldados adequadamente neste padrão. Paralelamente, diferentes bancos de dados NoSQL foram estudados com o propósito de buscar as ferramentas adequadas que atendessem as necessidades desejadas para validação da modelagem. No decorrer do estudo dos bancos de dados NoSQL foram identificados alguns limitadores dos sistemas chave-valor, como melhor adequação em um mapeamento de relacionamentos entre as entidades e o armazenamento apropriado para os diferentes tipos de dados. Por essa razão, essa família de NoSQL não foi utilizada neste trabalho. Sendo assim, as famílias de NoSQL utilizadas nesta dissertação foram: orientadas a documentos, família de colunas e orientadas a grafos. Foi realizado um estudo dos conceitos relacionados à Computação em Nuvem, com o objetivo de identificar características e serviços ofertados nesse ambiente.

Na sequência, foi estruturado um ambiente de execução em duas diferentes nuvens computacionais, Google Cloud e Digital Ocean. Essas duas plataformas foram escolhidas por oferecerem *vouchers* gratuitos de utilização dos serviços. Os sistemas NoSQL MongoDB, Cassandra e OrientDB e ferramentas da Bioinformática, utilizadas na execução das atividades dos *workflows*, foram instalados no ambiente de execução em cada uma das nuvens escolhidas. Os estudos de caso para validação da proposta utilizaram dois *workflows* reais da Bioinformática, *workflows* sequenciais fornecidos por biólogos da Universidade de Brasília.

Os *workflows* foram executados em cada uma das nuvens e os dados de proveniência foram coletados e armazenados de acordo com a modelagem proposta para cada um dos bancos de dados escolhidos, que foram configurados para operar em *cluster*. Os resultados das execuções foram avaliados segundo metodologia empírica. Foi possível armazenar, gerar grafos de proveniência e realizar consultas aos dados armazenados segundo a mode-

lagem proposta. Por meio dos resultados foi possível avaliar a viabilidade da modelagem proposta, possibilitando responder a questão de pesquisa inicialmente proposta.

Conforme os objetivos definidos, considera-se que a principal contribuição desta dissertação é a definição de uma modelagem de proveniência genérica para execução de experimentos científicos da Bioinformática em nuvem e que utiliza o PROV-DM. Foram utilizados três diferentes famílias de bancos de dados NoSQL para que a modelagem proposta pudesse ser validada nas execuções dos experimentos em nuvem. Para alcançar os objetivos descritos serão comparados os experimentos com trabalhos correlatos da literatura.

Conforme a metodologia descrita, esta dissertação foi estruturada da seguinte maneira:

- O Capítulo 2 apresenta a fundamentação teórica necessária para o desenvolvimento deste projeto, como *workflows* científicos, *workflows* de Bioinformática, Nuvens Computacionais, sistemas gerenciadores de bancos de dados, proveniência de dados e alguns trabalhos relacionados a esta pesquisa;
- O Capítulo 3 especifica o modelagem proposta seguindo o modelo PROV-DM para este trabalho e o ambiente de nuvem computacional utilizado;
- O Capítulo 4 apresenta os estudos de casos aplicados para validação da modelagem e as contribuições;
- O Capítulo 5 apresenta as conclusões do trabalho e os trabalhos futuros;

Capítulo 2

Fundamentação Teórica

Neste capítulo são tratados os conceitos pertinentes a *workflows* científicos, *workflows* de Bioinformática, nuvens computacionais, bancos de dados NoSQL e proveniência de dados. Assim, este capítulo foi dividido nas seguintes seções: a Seção 2.1 trata sobre os conceitos de *workflows* científicos, suas fases e sua utilização na área de Bioinformática; a Seção 2.2 descreve nuvens computacionais e seus conceitos e aplicações; a Seção 2.3 aborda os bancos de dados NoSQL e descreve os SGBD (Sistemas Gerenciadores de Banco de Dados) utilizados neste trabalho; a Seção 2.4 descreve a proveniência de dados, seus modelos e utilização em ambiente de nuvem computacional; e por fim, a Seção 2.5 mostra um resumo dos trabalhos encontrados e que são relacionados ao presente trabalho.

2.1 *Workflows* Científicos

Em tradução livre ao português, *workflow* traduz-se em "fluxo de trabalho". Em uma definição mais básica, *workflow* nada mais é que a composição de uma sequência bem definida de tarefas e suas dependências a serem executadas a fim de gerar um resultado específico. Um *workflow* científico pode ser definido como a especificação formal de um processo científico que representa os passos a serem executados em um determinado experimento (DEELMAN *et al.*, 2009). Essas tarefas ou passos podem ser programas ou sistemas que concebem a automatização a um processo, otimizando a forma de trabalho.

O processo de definição de um *workflow* científico não é independente do domínio de aplicação, pelo contrário, cada *workflow* requer uma ampla discussão entre os membros do grupo de pesquisa, sua definição geralmente envolve tomada de decisão e análises refinadas sobre cada uma se suas etapas (MATTOS *et al.*, 2008). A criação do experimento através da concepção dos *workflows* científicos necessita de processos de apoio, que permitam o reaproveitamento de partes de *workflows* anteriores e conhecimentos relacionados na construção de novos *workflows* (MATTOSO *et al.*, 2008).

Um *workflow* científico sequencial pode ser composto por diversas fases, dependendo de sua natureza. Cada fase pode ser configurada e parametrizada de acordo com um objetivo, e diferentes combinações de ferramentas podem ser utilizadas. Normalmente, essas fases são programas combinados e executados, e cada programa produz um conjunto de dados como saída conforme mostrado na Figura 2.1. O conjunto de dados resultante de um programa pode ser usado como dado de entrada para o próximo programa.

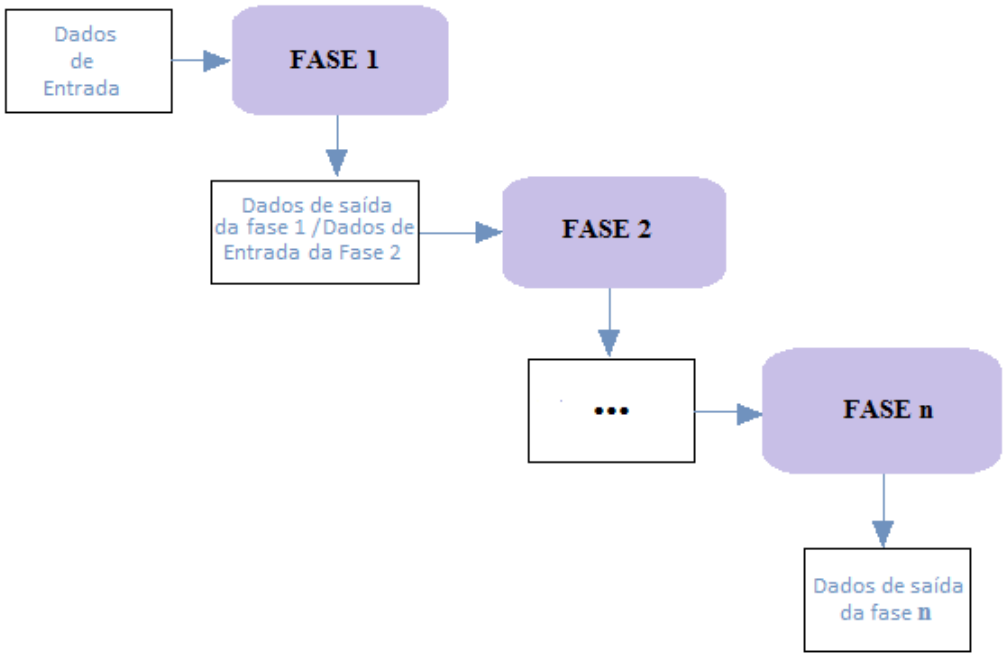


Figura 2.1: Exemplo de *Workflow* Composto de Dados de Entrada, "n" Fases e suas Saídas.

Uma das grandes vantagens de se utilizar um *workflow* científico é que o pesquisador pode focar somente no domínio de sua pesquisa e não se preocupar com trabalhos adicionais como, por exemplo, realizar a composição, a execução e a gerência dos programas, ou realizar a gestão dos dados manualmente, que despenderia tempo.

2.1.1 *Workflows* de Bioinformática

Diversas áreas da Biologia Molecular utilizam *workflows* em seus experimentos científicos (BOEKEL *et al.*, 2015), nos quais frequentemente são processados dados oriundos de projetos genoma, transcritoma, metaboloma, entre outros (WOLSTENCROFT *et al.*, 2013) (KOHL *et al.*, 2014). Cada execução de um *workflow* científico de Bioinformática pode gerar um grande volume de dados, os quais devem ser armazenados para novas execuções, análises ou confirmações de resultados.

Um dos problemas ao qual a Bioinformática se dedica é a montagem de fragmentos de DNA. Os fragmentos de DNA são oriundos do sequenciamento de alto desempenho e são chamados *reads*. As *reads* então são *strings* de um alfabeto que representa o DNA ou o RNA. A partir de alinhamentos das *reads*, a montagem obtém sequências contíguas (*contigs*) que representam o DNA original da amostra (ZERBINO; BIRNEY, 2008).

A montagem de fragmentos pode utilizar um genoma de referência, neste caso as *reads* são alinhadas contra um genoma de organismo filogeneticamente próximo ao organismo do qual provêm as *reads*. Já a montagem sem um genoma de referência é chamada de montagem *de novo* (BLEIDORN, 2017).

Experimentos científicos da Bioinformática geralmente são modelados como *workflows* científicos, que são utilizados especialmente em projetos genoma e transcriptoma, em experimentos que envolvem análise de sequenciamento de DNA e/ou RNA, como a montagem de fragmentos. De acordo com Saldanha (2012), as análises são necessárias pois os fragmentos produzidos pelos sequenciadores automáticos devem ter sua qualidade verificada, ser agrupados se os fragmentos forem muito pequenos ou ter identificadas suas funções biológicas, dentre outras. Essas análises podem ser realizadas em diferentes fases e em diferentes ferramentas que compõem os *workflows*.

Um típico *workflow* para montagem de fragmentos é normalmente composto por três fases sequenciais (HAAS *et al.*, 2013): Filtragem, Montagem/Mapeamento e Análise. A Figura 2.2 mostra um exemplo de *workflow* da Bioinformática. A característica sequencial dessas fases demanda que o resultado de saída de cada fase seja utilizado como entrada para a fase seguinte. Cada uma dessas fases é descrita a seguir:

- Filtragem: nesta fase, as SRS (*Short Reads Sequencing*) produzidas pelos sequenciadores de alto desempenho são filtradas através de parâmetros como, o grau de qualidade que possuem, se possuem regiões de mapeamento proveitosos para as fases que seguem no *workflow*, e outros filtros definidos na pesquisa. Os parâmetros utilizados para a filtragem variam, de acordo com a espécie estudada, com os objetivos do projeto e com a experiência dos pesquisadores (HUACARPUMA, 2012). Assim, as SRS de saída serão somente aquelas que atendam aos parâmetros estipulados pelo projeto.

- Montagem/Mapeamento: os dados de entrada desta fase são os arquivos de saída da fase de filtragem. As SRS filtradas são mapeadas a um genoma de referência - Figura 2.7 (a). Quando não há um genoma de referência, as SRS são agrupadas em uma sequência maior a fim de montar a sequência original - Figura 2.7 (b) - e assim as *reads* são alinhadas gerando um consenso. O conjunto de *reads* mais consenso é chamado de *contig*. A montagem é a fase na qual as SRS produzidas pelos sequenciadores automáticos são montadas de forma a reproduzir a sequência

genética original (DE PAULA, 2012). O arquivo de saída dessa fase é utilizado na fase de análise.

- Análise: nesta última fase é realizada a análise do resultado de toda a execução do *workflow*. É nesta fase que o pesquisador pode verificar se o resultado obtido foi positivo, ou seja, validar as suposições feitas inicialmente no experimento, ou negativo, se erros, no experimento ou na execução parcial ou total do *workflow*, foram encontrados.

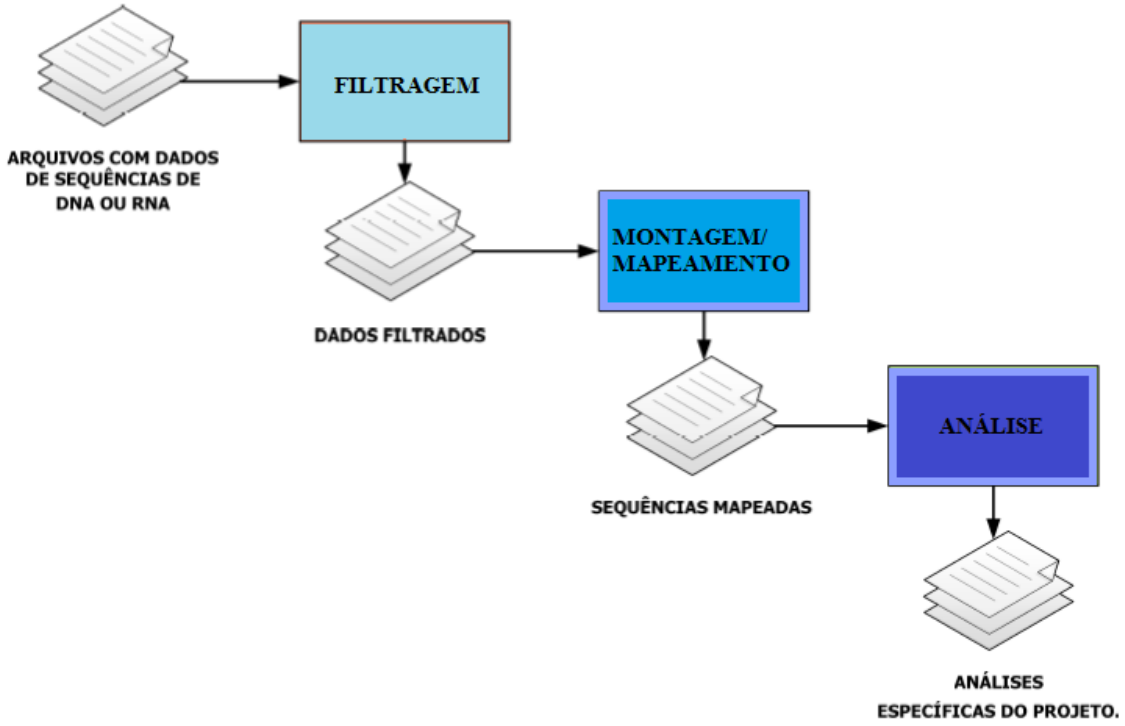
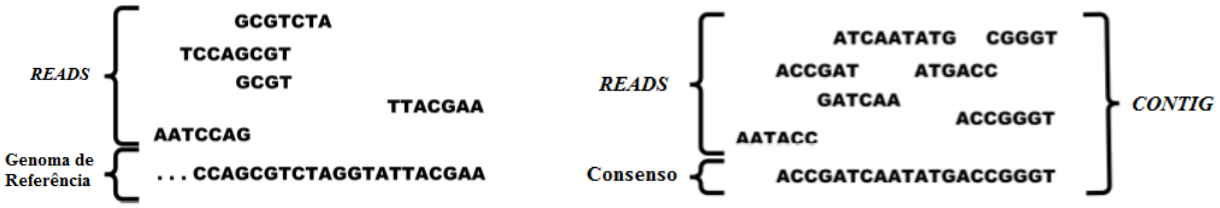


Figura 2.2: Exemplo de *Workflow* de Projetos Genoma e Transcriptoma.



(a) Processo de Mapeamento

(b) Processo de Montagem

Figura 2.3: Exemplo de um processo de mapeamento (a) e de um processo de montagem (b).

2.1.2 Ciclo de Vida de *Workflows* Científicos em Ambiente de Nuvem

A natureza das necessidades da computação científica se encaixa bem com a flexibilidade e a elasticidade, sob demanda, oferecida pelo paradigma de computação em nuvem (ASSIS, 2016). O ciclo de vida de um experimento científico pode ser composto por três fases: composição, execução e análise (MATTOSO *et al.*, 2010).

Já um *workflow* científico em ambiente de nuvem possui além destas três fases, mais a fase de configuração para adaptação do *workflow*. Assim, um *workflow* científico que é executado em ambiente de nuvem é composto pelas fases de Composição, Configuração, Execução e Análise, apresentadas na Figura 2.4.

Figura 2.4: Ciclo de Vida de um *Workflow* Científico em Nuvens (ASSIS, 2016).

Na fase de composição do *workflow* são feitas as especificações que impactam diretamente na configuração do ambiente de execução, tais como programas necessários, qual área da nuvem os dados se encontram, dentre outros. Em seguida, é realizada a fase de configuração do ambiente, na qual é feita a transferência de dados locais para a nuvem, criação e configuração do *cluster* virtual e outras configurações.

Como o tamanho do *cluster* pode aumentar ou diminuir ao longo do tempo, dependendo da demanda de processamento do *workflow*, existe um subciclo de redimensionamento, até que se finalize a execução do *workflow*.

Na fase de execução, as atividades do *workflow* são submetidas para as diversas máquinas virtuais, definidas na fase de configuração, e são executadas de forma distribuída. Desta forma, enquanto as atividades são executadas, é feito o monitoramento da execução do *workflow*.

A fase de análise é composta pelo *download* dos dados para uma máquina local, onde serão analisados, e pela subfase de descoberta, em que os pesquisadores farão suas verificações e conclusões sobre a hipótese de pesquisa original, ou seja, hipótese de pesquisa original foi corroborada ou refutada. Caso a hipótese seja corroborada, o ciclo é repetido para validar a hipótese. Caso seja refutada, o ciclo é repetido novamente com parâmetros de entrada diferentes.

2.2 Computação em Nuvem

A computação em nuvem é como uma infraestrutura de computação, provida sob demanda, que oferece comunicação e controle, sendo servida a partir da rede, de forma compartilhada e dinamicamente escalável (FOSTER *et al.*, 2008). Existem diversas definições para computação em nuvem, cada uma destacando características específicas de nuvens computacionais. De acordo com Armbrust *et al.* (2009), a computação em nuvem é definida como a união de aplicações oferecidas como serviço pela Internet com o hardware e o software localizados em *datacenters* de onde o serviço é provido. Esta definição, porém, é considerada por muitos como abrangente demais e que deixa de lado outras peculiaridades da computação em nuvem como o modelo *pay-per-use*.

Já Buyya *et al.* (2009) definem computação em nuvem como um modelo para oferta e utilização de recursos como serviços sob demanda, em um ambiente com múltiplos provedores, seguindo uma economia de escala. Esta definição deixa claro que o objetivo da computação em nuvem é proporcionar aos seus usuários uma sensação de existência ilimitada de recursos e que seus usuários devem pagar somente pelos recursos que forem utilizados (*pay-per-use*).

Computação em nuvem é um paradigma para provimento de serviços computacionais que vem criando um grande ecossistema formado por diferentes tecnologias e provedores de serviço, proporcionando aos usuários uma vasta variedade de opções de uso da nova tecnologia (SALDANHA, 2012).

Segundo Vaquero *et al.* (2008), Foster *et al.* (2008), as principais características existentes na computação em nuvem são:

- Flexibilidade: facilidade de agregação de novos recursos à medida que se tornem necessários;

- Elasticidade: disponibilizar e remover recursos computacionais em tempo de execução;
- Sob demanda: o usuário pode, conforme sua necessidade, requerer maior ou menor quantidade de recursos computacionais;
- Virtualização: virtualização de recursos como máquinas virtuais, virtualização de redes, de armazenamento de dados e de memória;
- Escalabilidade: aumento da capacidade de tarefas através da adição proporcional de recursos.

2.2.1 Arquitetura

O acesso e a disponibilidade de ambientes de computação em nuvem são definidos pelos modelos de implantação. Os modelos de implantação da nuvem existentes são detalhados a seguir (MELL; GRANCE, 2011):

- Nuvem pública: a infraestrutura da nuvem está disponível para o público em geral, sendo acessado por qualquer usuário que conheça a localização do serviço;
- Nuvem privada: a infraestrutura da nuvem é usada exclusivamente por uma organização. A nuvem pode ser de propriedade, gerenciada e operada pela própria organização, por terceiros ou uma combinação destes, e está regularmente localizada dentro de uma organização;
- Nuvem híbrida: é uma composição de duas ou mais nuvens (privadas, comunitárias ou públicas) que permanecem como entidades exclusivas, mas são agrupadas por tecnologia, permitindo a portabilidade de dados e aplicações;
- Nuvem comunitária: a infra-estrutura da nuvem é compartilhada em várias organizações e oferece suporte a uma comunidade específica que tenha os mesmos interesses ou restrições (por exemplo, missão, requisitos de segurança, políticas e considerações de conformidade). Pode existir localmente ou remotamente, e pode ser gerenciado e operado por uma ou mais organizações pertencentes à comunidade, por terceiros ou por uma combinação deles.

2.2.2 Modelos de Serviços

Além do modelo de implantação, existem os modelos de serviço, que definem um padrão arquitetural para o funcionamento das soluções baseadas em computação em nuvem. A Figura 2.5 detalha os diferentes modelos presentes na arquitetura. Estes modelos são definidos como (FOSTER *et al.*, 2008) (LOEFFLER, 2011):

- Software como Serviço (SaaS - *Software as a Service*): proporciona softwares com propósitos específicos que são disponíveis para os usuários através da Internet, e são acessíveis a partir de vários dispositivos por meio de uma interface como um navegador Web. No SaaS o usuário não administra ou controla a infraestrutura subjacente, incluindo rede, servidores, sistemas operacionais, armazenamento, ou mesmo as características individuais da aplicação, exceto configurações específicas. Exemplos de serviços oferecidos neste modelo: OpenNebula[1], Apache Hadoop[2].

- Plataforma como Serviço (PaaS - *Platform as a Service*): a PaaS oferece uma infraestrutura de alto nível de integração para implementar e testar aplicações na nuvem. Ela oferece linguagens de programação e ambientes de desenvolvimento para as aplicações, auxiliando a implementação de softwares. Os desenvolvedores dispõem de ambientes escaláveis, mas eles têm que aceitar algumas restrições sobre o tipo de software que se pode desenvolver, desde limitações que o ambiente impõe na concepção das aplicações, tais como linguagens de programação, bibliotecas, serviços e ferramentas suportadas pelo provedor. Exemplo de serviço deste modelo é o Google App Engine[3].

- Infraestrutura como Serviço (IaaS - *Infrastructure as a Service*): este modelo oferece a usuários comuns recursos como servidores, rede, armazenamento e outros recursos de computação para construir um ambiente de aplicação sob demanda (podendo haver cobrança baseada na utilização do serviço), que podem incluir sistemas operacionais e aplicativos. Em geral, o usuário não administra ou controla a infraestrutura da nuvem, mas tem controle sobre os sistemas operacionais, armazenamento e aplicativos implantados, e pode selecionar componentes de rede, tais como *firewalls*. Exemplo de serviço oferecido no modelo IaaS: Elastic Compute Cloud[4].

As aplicações oferecidas como serviço na camada SaaS podem ser desenvolvidas ou executadas pelas plataformas da camada PaaS, ou utilizar diretamente os recursos oferecidos pela camada IaaS (SALDANHA, 2012). Todas essas camadas são acessíveis ao usuário através da Internet.

2.3 Banco de Dados NoSQL

Os bancos de dados NoSQL (*Not only SQL*) surgiram em meados dos anos 90, a partir de uma solução de banco de dados, relacional e de código aberto, que não fornecia

[1]https://opennebula.org/
[2]http://hadoop.apache.org/
[3]https://cloud.google.com/appengine/?hl=pt-br
[4]https://aws.amazon.com/pt/ec2/

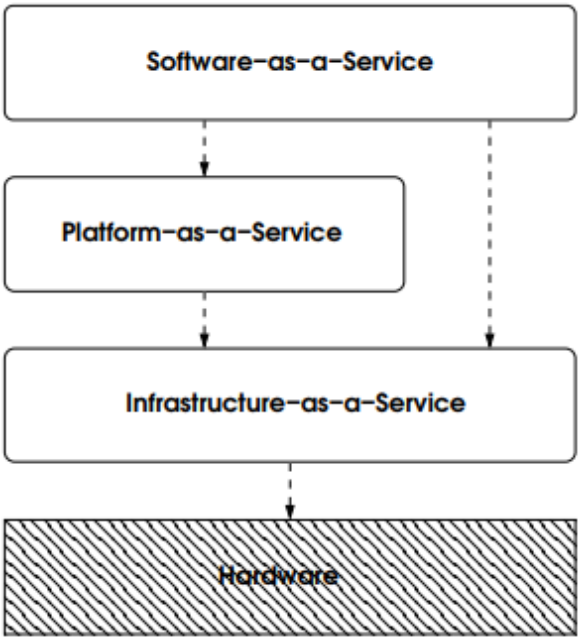


Figura 2.5: Modelos de Serviços Disponibilizados em Ambiente de Nuvem (SALDANHA, 2012).

uma interface SQL. Logo, emergiram como uma alternativa aos tradicionais Sistemas Gerenciadores de Bancos de Dados Relacionais (SGBDR) (MOHAMED *et al.*, 2014).

Posteriormente, esse termo passou a representar soluções que promoviam uma alternativa para o modelo relacional, introduzindo uma nova estratégia de armazenamento no qual os sistemas podem ter maior autonomia e flexibilidade de estruturação e ser livres de regras e características do modelo relacional.

Nos últimos anos, uma variedade de bancos de dados NoSQL foram desenvolvidos para atender as necessidades particulares de organizações com relação ao desempenho, escalabilidade e manutenção.

Estes sistemas são propostos como soluções escaláveis, contam com processamento distribuído, proporcionam alta disponibilidade, são flexíveis e têm capacidade para armazenar dados estruturados e não estruturados. Embora não exista um padrão único do que são os sistemas NoSQL, existem seis funcionalidades, de acordo com (CATTELL, 2011), que podem identificar estes sistemas:

- Escalabilidade horizontal e processamento distribuído;
- Transações, geralmente, não ACID (Atomicidade, Consistência, Isolamento e Durabilidade);
- Uso eficiente de índices distribuídos;

- Replicação e particionamento;
- Interface simples;
- Adicionar novos atributos dinamicamente aos registros.

2.3.1 Teorema CAP

O Teorema CAP (*Consistency, Availability, Partition Tolerance*) foi proposto por (BREWER, 2012) que definiu que para qualquer banco de dados distribuído existem três propriedades interdependentes básicas:

- Consistência (*Consistency*): um sistema é dito consistente se, após cada operação realizada, os dados permanecem consistentes, ou seja, uma leitura do banco de dados por parte de vários usuários resulta em uma mesma visão após uma atualização feita por um outro usuário;
- Disponibilidade (*Availability*): propriedade que garante que toda operação, ao seu término, assegure a disponibilidade do serviço, tenha ela sido bem sucedida ou não, em outras palavras, qualquer usuário terá sua requisição atendida a todo momento, através do compartilhamento e replicação dos dados;
- Tolerância à partição (*Partition Tolerance*): propriedade em que um sistema pode executar corretamente, mantendo todas as suas características independente da divisão física da rede, de perda de dados ou de falhas no sistema, ou seja, o sistema de banco de dados deve permanecer disponível ainda que partes do mesmo estejam inativas devido a algum problema.

O Teorema CAP diz que, dentre as três propriedades, só é possível utilizar duas ao mesmo tempo dentro de um sistema distribuído. A impossibilidade de usar mais de duas propriedades vem do fato de que, ao escolher duas delas para utilização, a outra é instantaneamente inviabilizada. Desta forma é necessário escolher apenas duas delas, de acordo com a funcionalidade da aplicação em questão. A Figura 2.6 mostra as possíveis relações entre as propriedades do Teorema CAP e as classificações destas relações em três categorias: CA - Consistência e Disponibilidade, CP - Consistência e Tolerância à partição e AP - Disponibilidade e Tolerância à partição.

2.3.2 Modelo de Dados

Na literatura existem várias classificações para os modelos de dados de sistemas gerenciadores de bancos de dados NoSQL, porém os quatro modelos mais citados são ((CORBELLINI *et al.*, 2017), (POURABBAS, 2014)):

Figura 2.6: Teorema CAP.

- Chave–valor: os dados são armazenados indexados por chaves, divididos em duas partes, onde cada valor (um tipo de dado) está associado a uma chave única (um identificador exclusivo). As operações de leitura só podem ser executadas através das chaves, não permitindo consultas mais complexas, e são limitadas ao resultado exato. É um modelo que possui alta velocidade em consultas. Exemplo: Redis[5].

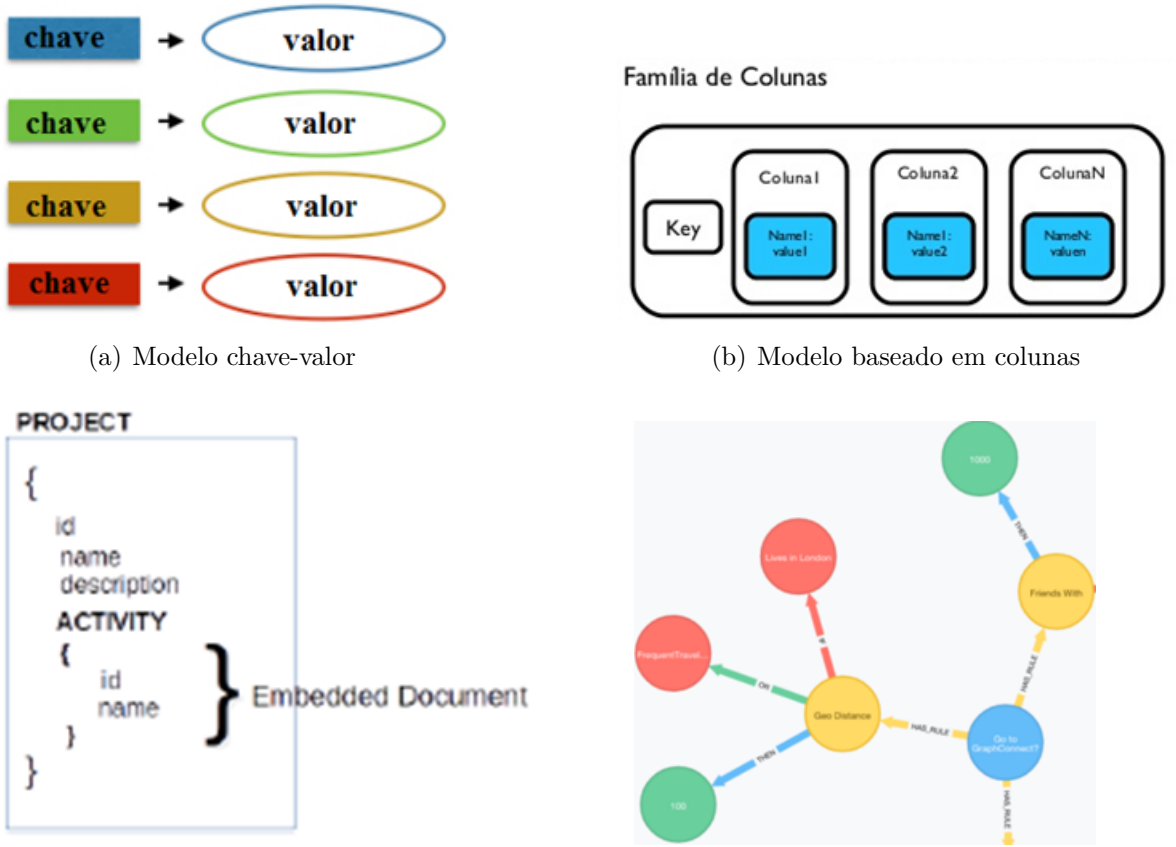
- Baseado em colunas: a estrutura de valores é definida como um conjunto de colunas predefinido. Também é conhecido como modelo tabular. Os registros podem ser armazenados particionados verticalmente e horizontalmente entre os nós. Nesse modelo, pode-se adicionar novas colunas de acordo com a necessidade do usuário e estas são organizadas em famílias de colunas para facilitar a organização e particionamento do banco de dados. Algumas das mais relevantes características deste modelo são a indexação, o particionamento e a alta compressão dos dados. Exemplo: Cassandra[6], HBase[7].

- Baseado em documentos: os documentos armazenados são coleções de atributos e valores, podendo conter atributos multivalorados. Eles utilizam o conceito de chaves e valores, onde cada documento contém uma chave que o identifica. Os documentos são codificados em um formato de dados padrão, como JavaScript Object Notation (JSON), eXtensible Markup Language (XML) ou Binary JSON (BSON). Exemplo: MongoDB[8].

- Grafos: os esquemas são representados por grafos direcionados ou não, em que os dados são armazenados nos vértices. Os relacionamentos são representados pe-

[5]https://redis.io/
[6]http://cassandra.apache.org/
[7]https://hbase.apache.org/
[8]https://www.mongodb.com/

las arestas que também podem armazenar dados dependendo do banco de dados. Exemplo: Neo4j[9].



(a) Modelo chave-valor

(b) Modelo baseado em colunas

(c) Modelo baseado em documentos

(d) Modelo baseado em grafos

Figura 2.7: Exemplo de um processo de mapeamento (a) e de um processo de montagem (b).

Alguns NoSQL são híbridos e implementam mais de uma família. O OrientDB[1], por exemplo, implementa as famílias Grafo, Chave–valor e Documento. Outro exemplo de NoSQL híbrido é o ArangoDB[2], que também implementa as famílias Documento, Grafo e Chave–valor.

Cada um destes modelos tem suas próprias características. Não existe um modelo considerado melhor que os outros. Porém, para resolver problemas específicos existem certos sistemas de bancos de dados mais apropriados. Desta forma, é necessário que se analise o problema para que se escolha a melhor ferramenta.

Neste contexto, a primeira tarefa para a definição dos sistemas gerenciadores de bancos de dados a serem utilizados neste trabalho foi a escolha do modelo de dados para

[9]https://neo4j.com/4j
[1]http://orientdb.com/
[2]https://arangodb.com/

armazenar os dados de *workflows* científicos de Bioinformática. Para abranger e avaliar diferentes tipos de modelos NoSQL, foram selecionados os modelos baseado em documentos, baseado em colunas e modelo de grafo. Nesses três modelos é possível armazenar tanto os dados brutos como também a proveniência de dados. O objetivo de selecionar esses três modelos foi de abranger as diferentes famílias dentre os sistemas gerenciadores de bancos de dados não relacionais para gerenciar dados de proveniência.

Os sistemas gerenciadores de bancos de dados NoSQL, baseados em documentos e em colunas, MongoDB e Cassandra, respectivamente, foram escolhidos para o estudo de caso desta dissertação. A escolha desses bancos de dados foi baseada no trabalho de Holt *et al.* (2015) e Guo *et al.* (2016), que destacaram que estes dois SGBD NoSQL estão entre os sistemas gerenciadores de bancos de dados mais utilizados atualmente. Já o OrientBD foi escolhido como a solução em modelo de grafo por ser um sistema distribuído e gratuito, permitindo a distribuição dos dados em *clusters*.

2.3.3 Cassandra

Surgiu em 2008, desenvolvido em Java, baseado no modelo de dados do BigTable do Google e na arquitetura do Dynamo da Amazon. Seu modelo de dados é orientado à família de colunas. É um sistema considerado altamente escalável, ou seja, mesmo que as requisições aumentem, seu desempenho é mantido.

Seu modelo é altamente desnormalizado e projetado para capturar e consultar dados rapidamente. Não existem conceitos de chaves estrangeiras, integridade referencial ou junções. Embora o Cassandra tenha objetos que se assemelham a um banco de dados relacional (por exemplo, tabelas, chaves primárias, índices), os dados não devem ser modelados utilizando o paradigma entidade-relacionamento, como em um banco de dados relacional. A sua modelagem de dados é feita através da compreensão de quais perguntas você precisará para consultar ao banco de dados (CHEBOTKO *et al.*, 2015) .

A unidade mais básica no modelo de dados do Cassandra é a coluna, formada por nome, valor e *timestamp*. Um *keyspace* é o agrupamento de dados que pode ser comparado a um esquema em um banco de dados relacional, porém seu diferencial é que possui informações de como os dados serão replicados ao longo de cada computador no *cluster* (LAKSHMAN; MALIK, 2010). A Figura 2.8 ilustra um *keyspace* do Cassandra. Uma família de colunas é composta por linhas e cada linha é formada por colunas e um atributo chave. Fazendo uma analogia a bancos de dados relacionais, o *keyspace* é equivalente ao banco de dados, a família de colunas equivale a uma relação (tabela) e uma coluna é comparada a um atributo. O Cassandra oferece uma linguagem de consulta própria chamada CQL (*Cassandra Query Language*), que se assemelha à linguagem SQL (HEWITT, 2010).

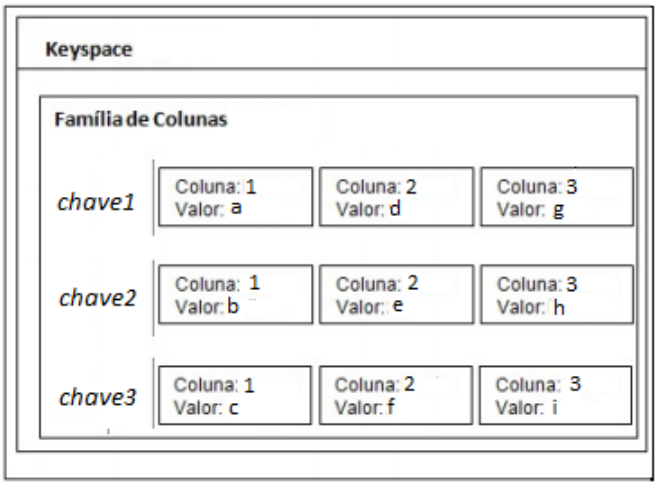


Figura 2.8: *Keyspace* do Cassandra

Algumas das características do Cassandra são ((LAKSHMAN; MALIK, 2010), (HEWITT, 2010), (ANICETO *et al.*, 2015), (LIMA, 2016)): distribuído, descentralizado, escalável, consistente, alta tolerancia a falhas, alta disponibilidade e alta performance.

2.3.4 MongoDB

O MongoDB é um sistema *open source* desenvolvido em C++ e lançado em 2009. Possui um modelo de dados orientado a documentos onde os dados são armazenados como uma representação binária do formato JSON (BSON). Cada documento armazenado é identificado por uma chave, a qual é uma sequência de caracteres (normalmente uma combinação do ID e um valor *timestamp*) e não pode ser duplicada (SINGH, 2015), tamanho máximo de 16 MB e pode conter diferentes esquemas (esquema flexível) (GUIMARAES *et al.*, 2015).

O MongoDB usa o conceito *master-slave*, amplamente utilizado em caso de falha de leitura ou escrita, e *sharding*, utilizado para a escalabilidade horizontal, o que garante a durabilidade dos dados (ABRAMOVA; BERNARDINO, 2013). No modelo de dados do MongoDB é possível organizar os dados como uma coleção de documentos com campos e valores padrão para cada registro armazenado. A Figura 2.9 ilustra a estrutura do MongoDB composto pelas coleções de documentos.

A linguagem de consulta usada pelo MongoDB é baseada em chamadas de API e JavaScript, entre outras (BOICEA *et al.*, 2012). Suas principais características são flexibilidade, facilidade de uso, velocidade de armazenamento, consistência, durabilidade e atomicidade condicional.

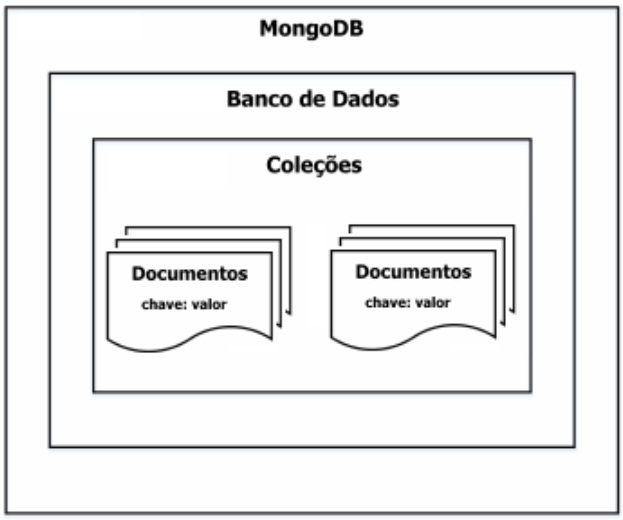


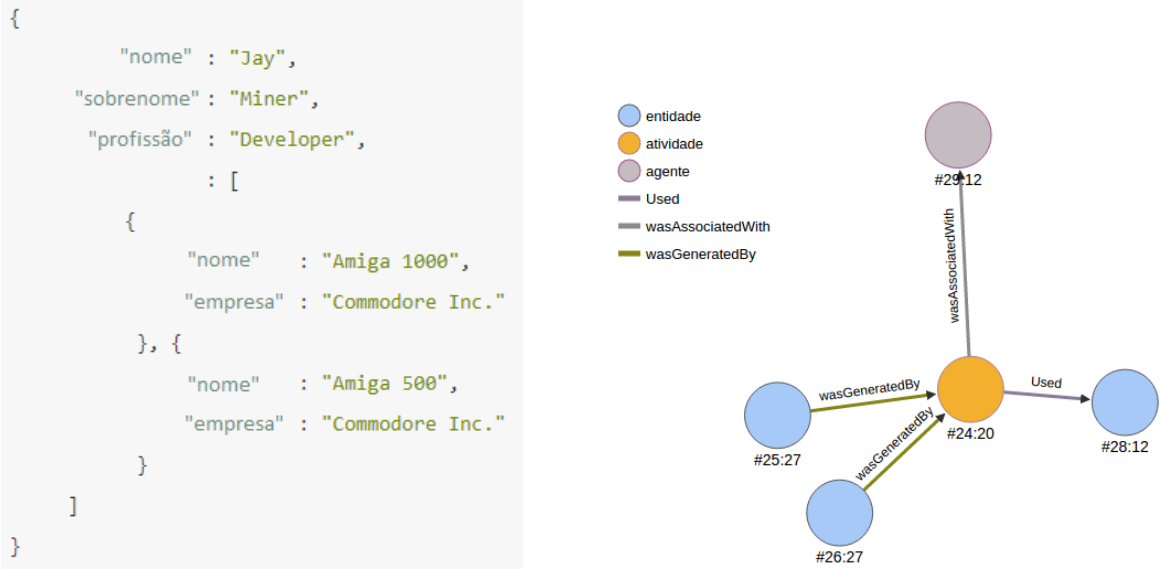
Figura 2.9: Estrutura do MongoDB.

2.3.5 OrientDB

Este sistema multimodelo foi lançado em 2010 como um projeto *open source* desenvolvido em Java. Ele é considerado multimodelo pois suporta abordagens orientadas a grafo, documento, chave-valor e objeto, combinando-as para reduzir a complexidade operacional e para manter a consistência dos dados. Os relacionamentos são gerenciados como em banco de dados em grafo, utilizando conexões diretas entre os registros (MOTA, 2016).

O modelo de grafo é uma estrutura semelhante à de uma rede consistindo em Vértices (também conhecidos como Nós) interconectados por Arestas (também conhecidos como Arcos). Já no modelo de documentos os dados são armazenados dentro de documentos. Um documento é um conjunto de pares de chave/valor (também chamados de campos ou propriedades), em que a chave permite acesso ao seu valor (TESORIERO, 2013). A Figura 2.10(a) ilustra o modelo de documentos e a Figura 2.10(b) o modelo de grafo do OrientDB.

A linguagem de consulta do OrientDB é o SQL e possui também algumas extensões para manipular árvores e grafos. Possui duas versões, a empresarial e a comunitária, cada uma com suas funcionalidades como, replicação para a versão comunitária e suporte técnico disponível 24 horas para a versão empresarial. Principais características deste sistema são robustez, altamente escalonável e distribuído.



(a) Modelo de documento do OrientDB

(b) Modelo de grafo do OrientDB

Figura 2.10: Modelos de documento e grafo do OrientDB.

2.4 Proveniência de Dados

O significado de proveniência é origem ou procedência. A definição mais relevante foi descrita por Buneman *et al.* (2001) que descreveu o termo proveniência como ”linhagem” ou ”*pedigree*” que referencia o histórico de como aquele dado foi produzido ou derivado. Segundo Almeida (2015), a proveniência de dados vem se tornando cada vez mais presente no ambiente científico, tanto para garantir a origem dos dados como para avaliar a sua acurácia. Segundo de Paula (2012), a proveniência permite aos cientistas estudarem detalhes de seus experimentos, e sempre que necessário reexecutá-los de uma forma mais planejada e controlada.

A utilidade da proveniência de dados vai muito além da reprodução de experimentos, segundo Marinho *et al.* (2009) a procedência é útil pois fornece aos cientistas uma variedade de aplicações de análise de dados, permitindo, por exemplo, verificar a qualidade dos dados gerados através da análise de suas referências ancestrais e determinar se são confiáveis ou não.

Algumas funcionalidades, segundo Goble (2002), da proveniência de dados são:

- Qualidade dos Dados: através do histórico de todo processo de criação do dado ou execução do experimento, de quem gerou, de qual base de dados em que o dado foi armazenado e etc., é possível estimar o grau de qualidade e confiabilidade daquele dado utilizado;

- Controle de replicação: a proveniência detalhada permite que um dado/experimento seja replicado por meio dos mesmos procedimentos, mesmas ferramentas e com os mesmos parâmetros;
- Propriedade e segurança: contém o controle sobre o dono do experimento e todos seus dados tanto para fins de direitos autorais e citações como também para responsabilidades caso os dados estejam errados;
- Informacional: informações importantes para a pesquisa são capturadas na proveniência como, o autor, membros da equipe, local e etc., que promovem um contexto relevante para a interpretação dos dados.

De acordo com Guimarães e Cavalcanti (2009), a proveniência pode ser capturada por duas formas: prospectiva e retrospectiva. A prospectiva captura os passos que devem ser seguidos para a geração de um dado, por exemplo, dados pertinentes à estrutura do *workflow* e configurações de ambiente utilizadas para sua execução. Já a retrospectiva captura os passos que foram executados para gerar um dado, por exemplo, informações sobre as máquinas virtuais envolvidas na execução das atividades de um derterminado *workflow*. Essas duas formas de captura são independentes, ou seja, não é necessário realizar a retrospectiva para se realizar a captura prospectiva.

2.4.1 *Provenance Data Model* (PROV-DM)

O PROV-DM é um modelo genérico que teve a sua primeira versão desenvolvida em 2011 sendo uma recomendação padrão do W3C (*World Wide Web Consortium*). Permite que diferentes sistemas importem e exportem suas representações específicas da proveniência entre si. Possui maior detalhamento e permite demonstrar a proveniência de maneira mais precisa. Objetiva descrever pessoas (agentes), entidades e atividades envolvidas na geração de um dado (W3C, 2018) através de um grafo direcionado. Neste grafo, a raiz representa a entidade cuja proveniência está sendo representada e as arestas que saem dela são direcionadas para as atividades e entidades das quais foi originada. A Figura 2.11 ilustra os símbolos utilizados para representar os diferentes nós do grafo no PROV-DM.

Esse modelo possui oito componentes que detalham seus elementos e relações entre si (W3C, 2018):

- Entidades (*Entities*): que representam qualquer objeto;
- Atividades (*Activities*): que representam os processos que utilizam e geram as Entidades;
- Agentes (*Agents*): são Entidades que influenciam as execuções das Atividades;

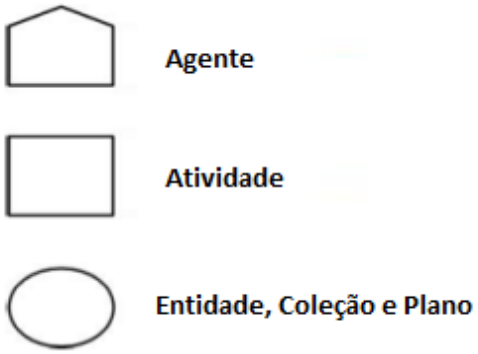


Figura 2.11: Representação Gráfica dos Nós do Modelo PROV-DM (W3C, 2018).

- Derivações (*Derivations*): descrevem a relação entre diferentes Entidades, permitindo demonstrar a dependência entre as Entidades usadas e geradas;
- Coleções (*Collections*): são coleções de Entidades que podem ter a sua proveniência demonstrada de forma coletiva;
- Anotações (*Annotation*): mecanismos de anotações para os elementos do modelo.
- Plano (*Plan*): conjunto de ações que um Agente deve seguir para alcançar um objetivo;
- Conta (*Account*): conjunto de informações que compõe um grafo de proveniência.

Existem diversos relacionamentos entre os componentes deste modelo, porém, para esta dissertação, foram considerados os mais relevantes identificados no contexto da Bioinformática. A Figura 2.12 mostra alguns dos possíveis relacionamentos existentes entre os elementos:

- *used*: indica que uma Entidade foi usada por uma Atividade;
- *wasGeneratedBy*: indica que uma Entidade foi gerada por uma Atividade;
- *wasAttributedTo*: atribui algum tipo de responsabilidade a um Agente sobre uma Entidade;
- *wasAssociatedWith*: atribui algum tipo de responsabilidade a um Agente sobre uma Atividade;
- *memberOf*: indica que uma Entidade particular é um membro de uma Coleção;
- *wasDerivedFrom*: indica que uma Entidade original foi usada, direta ou indiretamente, para gerar outra Entidade derivada.

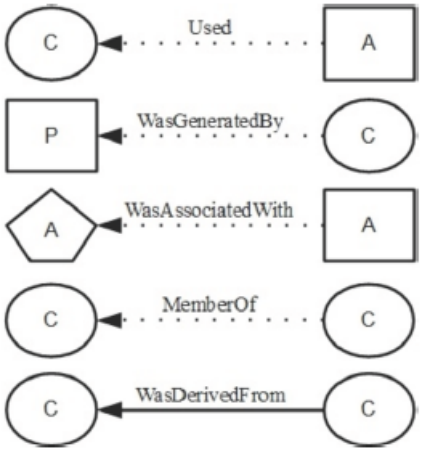


Figura 2.12: Relações entre os nós do modelo PROV-DM (W3C, 2018).

Anotações

O modelo PROV-DM oferece um recurso para adição de anotações (informações extras) no grafo de proveniência por meio de um identificador chamado Nota (*Note*). Este identificador representa um conjunto de pares atributo-valor que permite-se criar diversos tipos de anotações. Estas Notas podem ser conectadas tanto a nós quanto às arestas do grafo. Quando conectada a uma aresta, outra relação é criada, chamada de *hasAnnotation*. A Figura 2.13 mostra um exemplo de grafo baseado no modelo PROV-DM, onde podem ser vistas anotações.

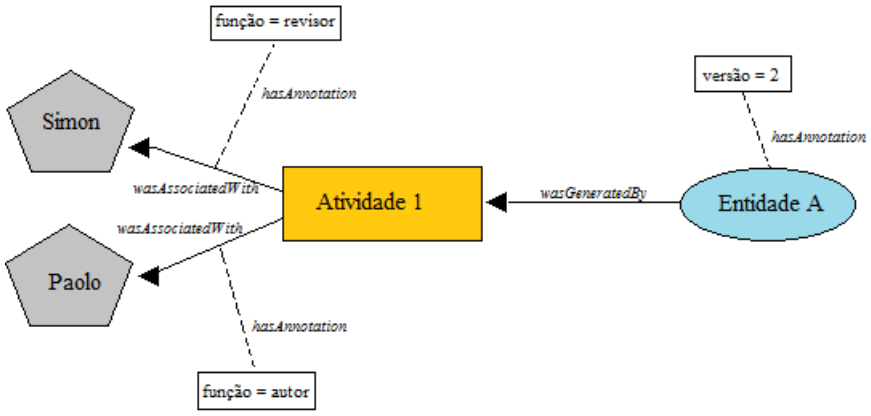


Figura 2.13: Anotações presentes em um grafo de proveniência do PROV-DM. Adaptado de (W3C, 2018).

2.4.2 Proveniência de Dados em Bioinformática

A Bioinformática é uma área multidisciplinar que faz uso intenso de ferramentas computacionais e, segundo Mattos *et al.* (2008), tem como objetivos finais a coleta, a organização, o armazenamento, a recuperação e as análises de dados biológicos, propiciando a inferência ou descoberta de informações sobre a biologia e/ou evolução dos organismos.

Na Bioinformática os experimentos podem ser realizados por diversas fases, executando diferentes programas com configurações específicas e parâmetros, por diferentes equipes e processando uma grande quantidade de dados. Segundo de Paula (2012), manter a proveniência de dados em projetos de bioinformática requer uma solução que permita armazenar a ligação entre os dados processados, juntamente com as informações das execuções de cada processo e de seus resultados.

Na execução de experimentos científicos, uma informação muito importante de proveniência na Bioinformática é o usuário, atuante principal da execução do experimento, bem como sua função exercida ao longo da pesquisa. Isso porque, nos laboratórios, diversos usuários podem trabalhar em diferentes pesquisas e essa informação pode influenciar tanto na confiabilidade quanto na qualidade do trabalho. Também é importante para os participantes de projetos da Bioinformática a visualização da proveniência de forma simples e objetiva através de interfaces adequadas e intuitivas, que mostrem as informações e suas relações de dados.

Junto a pesquisadores e biólogos, de Paula (2012) fez um levantamento e definiu em seu trabalho informações de proveniência relevantes aos experimentos científicos da Bioinformática. Os elementos principais retratados na proveniência foram:

- Projeto: informações sobre um projeto específico da Bioinformática, como, nome do projeto, descrição, instituições financiadoras e participantes, coordenador, data inicial e final do projeto;
- Experimento: informações sobre determinado experimento dentro do projeto, que são, nome do experimento, descrição, local de execução, data inicial e final da execução, anotações adicionais e número e data da versão gravada;
- Agente: informações dos usuários que influenciam as atividades de um experimento, tais como: nome do usuário, instituição do usuário, cargo ou função e anotações adicionais;
- Atividade: representa os processos executados que originaram o objeto foco da proveniência, e suas principais informações são: nome da atividade, nome do programa, versão do programa, linha de comando com os parâmetros utilizados, data e hora de

início e fim da atividade, descrição do ambiente computacional em que a atividade rodou e informações adicionais;

- Entidade: arquivos utilizados no experimento, tendo como informações principais o nome da entidade, descrição, localização do arquivo ou banco de dados e anotações adicionais;
- Coleção: informações de coleção de entidades do experimento, tais como: nome da coleção, número de entidades contidas na coleção, descrição, localização, e anotações adicionais.

2.4.3 Proveniência de Dados em Ambientes Distribuídos

Workflows executados em ambientes distribuídos, como as nuvens de computadores, podem aumentar a quantidade de recursos computacionais usados em sua execução. E isso também aumenta a quantidade de dados de proveniência a serem gerenciados (FERREIRA *et al.*, 2014). De acordo com Marinho *et al.* (2009), existem vários cenários de execução de *workflows*, em ambiente distribuído, classificados em quatro tipos:

- Execução remota de uma ou mais atividades de *workflow*;
- Execução remota de um sub-*workflow*;
- Execução remota de um sub-*workflow* por outro SGWfC(Sistema Gerenciador de Workflow Científico); e
- Execução de um ou mais *workflows* que fazem parte do mesmo experimento em SGWfC distintos.

Cada cenário possui características peculiares na execução do *workflow*, onde, dependendo do ambiente, um ou mais SGWfC diferentes realizam a gerência das informações de proveniência, centralizada ou distribuída, tornando mais complexo o processo de análise para o pesquisador.

A administração e a transparência de acesso a essa grande quantidade de dados de proveniência torna-se um grande desafio. Segundo Mattoso *et al.* (2008), o problema se agrava quando o experimento científico ocorre de modo distribuído e em larga escala, fazendo-se necessário um sistema que gerencie a composição de processos e dados num *workflow* científico, e que registre as etapas realizadas com as escolhas de parâmetros de execuções bem sucedidas do experimento, independente do local de execução. Os dados de proveniência necessitam gravar com precisão quem realizou qual atividade, qual atividade gerou qual dado e onde ele se encontra armazenado.

Normalmente, durante a execução distribuída de um *workflow*, diferentes sistemas dispersos geograficamente podem ser utilizados, demandando maior gerenciamento e eficácia tanto nas diversas consultas aos dados de proveniência, realizadas durante a execução do *workflow*, quanto na armazenagem dos dados gerados na execução.

Para que esse problema seja solucionado, os sistemas gerenciadores necessitam ter funcionalidades comuns em sistemas distribuídos como alta escalabilidade, e necessitam levar em consideração preocupações características de execuções distribuídas em ambientes de alto desempenho.

Recentemente, trabalhos de disponibilização de dados de proveniência, de execuções distribuídas e em tempo de execução, estão em destaque, principalmente, pelo fato de que *workflows* executados em nuvem podem ser complexos e executar por dias ou até semanas, demandando e alocando recursos pagos. Caso haja algum problema de execução, o pesquisador poderá prevenir-se a tempo de que ocorra uma sequência de erros e que recursos financeiros desnecessários sejam gastos.

2.5 Trabalhos Relacionados

Nesta seção são apresentados alguns trabalhos relacionados ao projeto proposto, que abordam de maneira semelhante a proveniência de dados em ambientes distribuídos.

Muniswamy-Reddy e Seltzer (2010) analisaram o impacto do uso dos dados de proveniência em nuvem e suas vantagens em relação às necessidades desses ambientes como, otimização de pesquisas, detectar anomalias em aplicações e controle de acesso. Utilizaram o modelo OPM para descrever o uso da proveniência em serviços disponibilizados pelos provedores de nuvem computacional para aumentar o valor dos serviços disponibilizados. Também realizaram uma análise do Amazon S3 com o SimpleDB, Microsoft Azure Blob Storage em conjunto com o Azure Table Service e o Nirvanix Internet Media File System em relação ao armazanamento da proveniência.

O trabalho proposto por Zhang *et al.* (2011) aborda a proveniência em ambiente de nuvem, para confiabilidade em provedores de serviços em nuvem, dividida em camadas de granularidades com base no tipo de recursos utilizados. Cada granularidade levantada possui suas informações de proveniência pertinentes aos recursos que utiliza, são elas: Aplicação, Máquina Virtual, Máquina Física, Nuvem e Internet. Também é proposta a ferramenta DataPROVE, que visa permitir coleta e armazenamento de proveniência em nuvem para melhor integridade, segurança, transparência e responsabilidade de dados gerenciados em uma nuvem. A proveniência é capturada através de logs em camadas do sistema e de aplicativos que são utilizados em uma nuvem.

Santos *et al.* (2013) realizaram consultas em dados distribuídos de proveniência durante a execução em tempo real de *workflows* em ambiente de nuvem. Para o trabalho foi utilizado o SGWf científico SciCumulus (OLIVEIRA *et al.*, 2010) que segue o PROV-Wf (W3C, 2018). O objetivo deste trabalho era analisar e avaliar o comportamento das consultas aos dados de proveniência distribuídos, em tempo real, em relação às consultas em base de dados centralizados.

Ferreira *et al.* (2014) fizeram um estudo comparativo de desempenho da gerência de dados de proveniência entre o SGBD relacional PostgreSQL e o SGBD NoSQL Cassandra. O estudo de caso foi feito com o *workflow* de Bioinformática SciPhy (OCAÑA *et al.*, 2011), que executa uma varredura de parâmetros em um conjunto de sequências de DNA, RNA ou aminoácidos e gera diversas árvores filogenéticas que mostram o quanto um organismo se assemelha a outro evolutivamente, utilizando o SGWf SciCumulus para análise do impacto na migração da base de dados.

A arquitetura distribuída Matriohska, foi desenvolvida por Cruz *et al.* (2014), implementada para capturar diferentes tipos de metadados de proveniência prospectiva e retrospectiva de *workflows* executados em ambiente de nuvem pública e armazenar seguindo o características do modelo PROV. No trabalho foi utilizado o *workflow* genômico OrthoSearch, desenvolvido para análises de grandes volumes de dados biológicos em estudos sobre doenças tropicais, e o SGWfC VisTrails como apoio a implementação dos *workflows*. O propósito era avaliar o desempenho da ferramenta quanto ao tempo de execução e a viabilidade da coleta de proveniência.

Assis (2016) implementou uma ferramenta entitulada ProvDoop e que realiza a captura e o armazenamento de dados de proveniência de execuções de *workflows* científicos na nuvem com o Hadoop. O ProvDoop foi implementado para ser acoplado ao Hadoop, capturar e armazenar a proveniência no banco de dados PostgreSQL configurado em uma máquina dedicada no *Amazon Relational Database Service* (Amazon RDS), sendo possível realizar consultas em tempo de execução. O objetivo do trabalho era avaliar os impactos da execução do ProvDoop comparado ao desempenho das execuções de *workflow* sem proveniência e determinar se a solução proposta era viável.

Oliveira *et al.* (2017) propuseram uma taxonomia com o objetivo de fornecer definição e classificação comuns para a reprodutibilidade das pesquisas. Foram propostos diversos requisitos como a proveniência da execução da experiência, assim como ambiente de nuvem computacional, que podem desempenhar um papel fundamental para que experimentos possam ser reproduzidos. Também criaram uma arquitetura genérica para experimentos em nuvem que propõe auxiliar na caracterização de ferramentas para a reprodutibilidade de experimentos.

de Paula (2012) propôs a utilização do modelo PROV-DM para representar a proveniência em projetos de Bioinformática. Uma comparação entre diferentes abordagens de modelagem de proveniência é feita a fim de analisar e elucidar que a proposta é vantajosa. Para validar o trabalho, foi desenvolvido o simulador de proveniência Provenance, em que o cientista é capaz de criar seus experimentos e mapear a proveniência, incluindo e armazenando dados para análise.

E por fim, é proposto por Almeida (2015) a modelagem PROV-DM e a utilização do banco de dados não relacional Neo4J, que é um banco de dados orientado a grafos que permite o armazenamento de entidades representadas por nós e seus relacionamentos, em uma arquitetura para captura automatizada da provêniencia de dados no contexto de *workflow* de Bioinformática. A arquitetura desenvolvida foi validada por meio da execução de *workflows* reais de Bioinformática.

A Tabela 2.1 é um resumo dos trabalhos relacionados apresentados anteriormente comparado com o proposto nesta dissertação. Note que foram comparados o ambiente de execução dos experimentos científicos (Nuvem Computacional), o modelo de proveniência utilizado, se o domínio de conhecimento é Bioinformática, e o(s) sistema(s) gerenciador(es) de banco de dados utilizado(s).

É importante ressaltar que nenhum dos trabalhos tem como objetivo a modelagem e o armazenamento de dados de proveniência de execuções de *workflows* científicos em nuvem em diferentes famílias de bancos de dados não relacionais: documento, família de colunas e grafo. O armazenamento da proveniência é no modelo PROV-DM, que utiliza como padrão um grafo acíclico como estrutura dos dados, e a modelagem traz informações pertinentes ao ambiente de nuvem.

Tabela 2.1: Trabalhos Relacionados.

Autores	Nuvem Computacional	Modelo de Proveniência	Dados Biológicos	Banco de Dados
Muniswamy-Reddy e Seltzer (2010)	X	OPM	-	-
Zhang *et al.* (2011)	X	-	-	-
Santos *et al.* (2013)	X	PROV-Wf	X	PostgreSQL
Ferreira *et al.* (2014)	X	PROV-Wf	X	PostgreSQL, Cassandra
Cruz *et al.* (2014)	X	PROV	X	MySQL
Assis (2016)	X	OPM	X	PostgreSQL
de Paula (2012)	-	PROV-DM	X	XML
Almeida (2015)	-	PROV-DM	X	Neo4J
Este trabalho	X	PROV-DM	X	MongoDB, Cassandra, OrientDB

Capítulo 3

Modelagem de Proveniência de Dados Proposta

Este capítulo traz a explanação das informações pertinentes sobre a modelagem da proveniência proposta para esta dissertação. Assim, a Seção 3.1 trata sobre a definição dos elementos de ambientes de nuvem computacional, a modelagem conceitual genérica proposta para esta dissertação. Em seguida, a Seção 3.2 trata sobre as modelagens definidas para cada banco de dados NoSQL, MongoDB, Cassandra e OrientDB.

3.1 Definição de Elementos do Ambiente de Nuvem

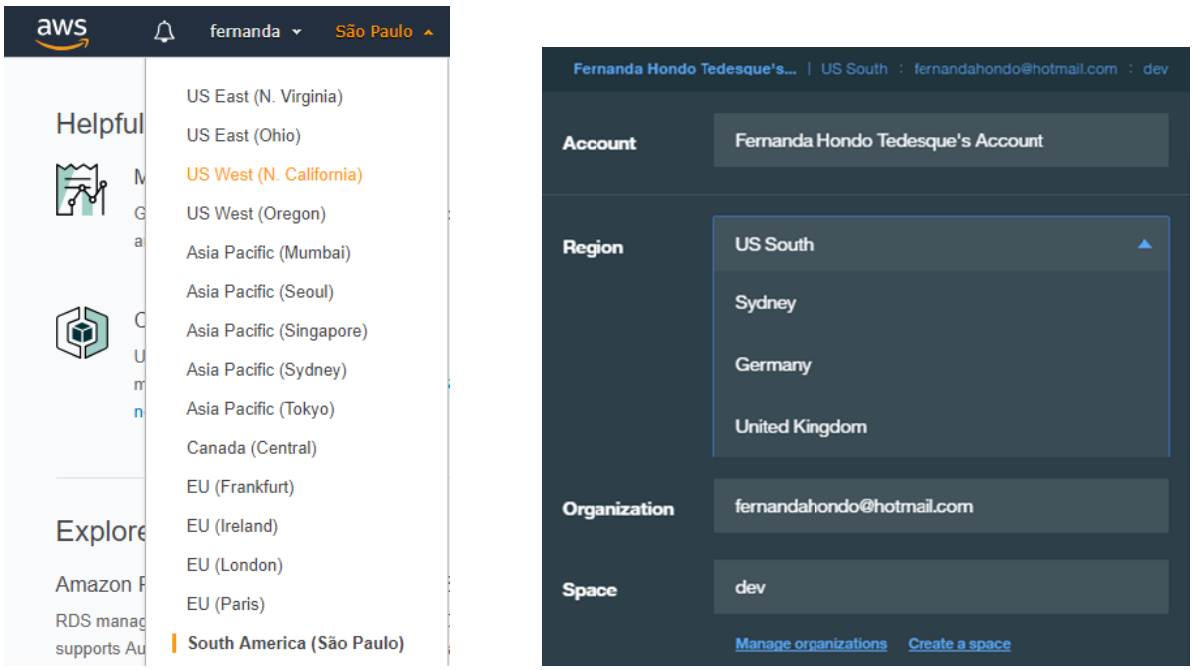
As características relacionadas a execução dos experimentos nas nuvens computacionais foram tratadas aqui neste trabalho. Para definir os elementos e informações necessárias a serem armazenadas sobre o ambiente de nuvem computacional, um estudo sobre como iniciar, configurar e utilizar projetos em ambiente de nuvem foi realizado, utilizando como alvos os provedores de nuvem mais populares que oferecem recursos fundamentais para a execução de projetos: Amazon[1], Azure[2], IBM BlueMix[3], DigitalOcean[4] e Google Cloud Platform[5].

Quando um usuário decide executar um projeto em ambiente de nuvem, a primeira questão a ser analisada é a escolha do provedor de nuvem, responsável por entregar os serviços a serem utilizados na execução dos seus experimentos.

Após o usuário decidir o provedor, ele escolhe a localização em que deseja executar seu projeto, logo, esta também foi uma informação considerada importante para a prove-

[1]https://aws.amazon.com/
[2]https://azure.microsoft.com/pt-br/
[3]https://www.ibm.com/cloud-computing/bluemix/pt
[4]https://www.digitalocean.com/
[5]https://cloud.google.com/?hl=pt-br

niência. Notou-se também que a escolha da região influencia diretamente na latência de comunicação entre as máquinas e que o preço e a disponibilidade dos serviços são diferentes por região. A Figura 3.1(a) mostra a tela inicial apresentada para o usuário quando ele vai iniciar seu projeto na Amazon. É possível observar que diferentes localizações e regiões são apresentadas para o usuário escolher. Já a Figura 3.1(b) detalha a escolha da localização no provedor da IBM BlueMix. Dessa forma, essas informações também foram consideradas relevantes para a proveniência de dados no ambiente de nuvem.



(a) Escolha da localização no provedor Amazon

(b) Escolha da localização no provedor BlueMix

Figura 3.1: Escolha da Localização do Projeto.

A configuração da máquina virtual onde as atividades dos experimentos serão executadas também é um passo importante. Cada provedor possui configurações de instâncias padrão, sendo possível aumentar ou diminuir os recursos de acordo com o que o usuário necessita. Essas configurações implicam diretamente no custo das instâncias. Outro elemento não menos importante é o *cluster* onde estarão executando as máquinas virtuais envolvidas no projeto de forma distribuída.

Assim, o resultado do estudo mostrou elementos que são comuns a todos os provedores, sendo que alguns deles são elementos definidos inicialmente, antes de iniciar qualquer projeto pelo usuário no provedor.

A Tabela 3.1 detalha as informações pertinentes a proveniência de dados de *workflow* de bioinformática em um ambiente de computação em nuvem. Como pode ser observado, foram levantados 4 elementos: Provedor, *Cluster*, Local e Máquina.

Tabela 3.1: Elementos da Proveniência em Nuvem.

Provedor		*Cluster*	
Nome	Nome do provedor	Nome	Nome do cluster
url	endereço eletrônico do provedor	Num_Maquinas	Quantidade de máquinas do cluster
Descrição	Descrição do provedor	Descrição	Descrição do cluster
Local		Maquina	
Região	Localidade. Ex :SP, NY, UE...	Hostname	identificação
Zona	Especificaçãoda região. Ex : zona1, zona2..	IP	ipAddress da máquina
		Tipo	tipo de configuração da maquina no provedor
		SistemaOperacional	sistema operacional rodando na maquina;
		Cpu	identificação do processador
		Ram	memória RAM da maquina
		Disco	quantidade de disco na maquina
		Tipo_disco	tipo do disco rígido da máquina
		Preço	preço da maquina no provedor
		Tipo_preço	especificação do pagamento : por hora, por uso, etc...

Todos esses elementos são cruciais para a proveniência de um experimento a ser executado em nuvem, uma vez que através dessas informações é possível, além de rastrear a origem do dado na nuvem, reproduzir um experimento em condições semelhantes do ambiente originalmente utilizado para o experimento.

3.1.1 Nuvens Computacionais Consideradas Neste Trabalho

Para esta dissertação, após o levantamento de informações e estudo sobre os diferentes provedores, foram considerados os provedores Google Cloud e DigitalOcean devido à gratuidade de serviços oferecida para novos usuários e *vouchers* de desconto na execução de instâncias.

Google Cloud Platform

O Google Cloud Platform consiste em um conjunto de ativos físicos, como computadores e unidades de disco rígido, além de recursos virtuais, como máquinas virtuais VM, contidos em *data centers* do Google em todo o mundo (GOOGLE, 2017). Nele, os recursos ou produtos são disponibilizados como serviços. Os usuários podem combinar serviços para obterem a estrutura necessária ao seu projeto. Além disse, ele fornece tipos diferentes de máquinas virtuais, com diferentes configurações.

No Google Cloud Platform um novo usuário recebe US$ 300 de crédito para usar por um período de 12 meses em todos os produtos do Google Cloud Platform como avaliação gratuita. O procedimento para criar uma instância no Google Cloud Platform segue descrição abaixo:

- Definição da localização da instância (região e zona), que implica em valores de custo diferente para cada região;
- Definição do tipo de máquina (núcleo, memória e GPUs): é possível escolher entre Standard, High Memory, High CPU, Small ou Micro, tamanho de memória que varia de 0,6GB a 1433.6GB;
- Definição do Disco de inicialização: Neste passo é possível escolher o sistema operacional da instância, aplicativos já configurados, tipo de disco de inicialização e o tamanho.

A Figura 3.2 mosta que a escolha da localização é feita no momento da criação de uma instância no provedor, assim como outras informações e configurações das instâncias a serem utilizadas no projeto.

Nuvem Computacional - DigitalOcean

DigitalOcean é uma empresa que tem como objetivo construir soluções de infraestrutura nas nuvens. Essa plataforma oferece suporte para que as equipes possam criar, implantar e dimensionar aplicativos em nuvem com mais rapidez e eficiência. Oferece tipos diferentes de máquinas virtuais, com diferentes configurações. Assim como no Google Cloud Platform, o procedimento para criar uma instância no DigitalOcean segue o seguinte conjunto de passos:

- Escolha do tipo da imagem da máquina com o sistema operacional, o tamanho da memória RAM, CPU e espaço de armazenamentos;
- Escolha da localização das máquinas a serem instanciadas, sendo região e zona;
- Configurações opcionais como *backup* e monitoramento.

A Figura 3.3 mosta o momento da criação de uma instância no provedor, as configurações das instâncias a serem utilizadas no projeto e a localização.

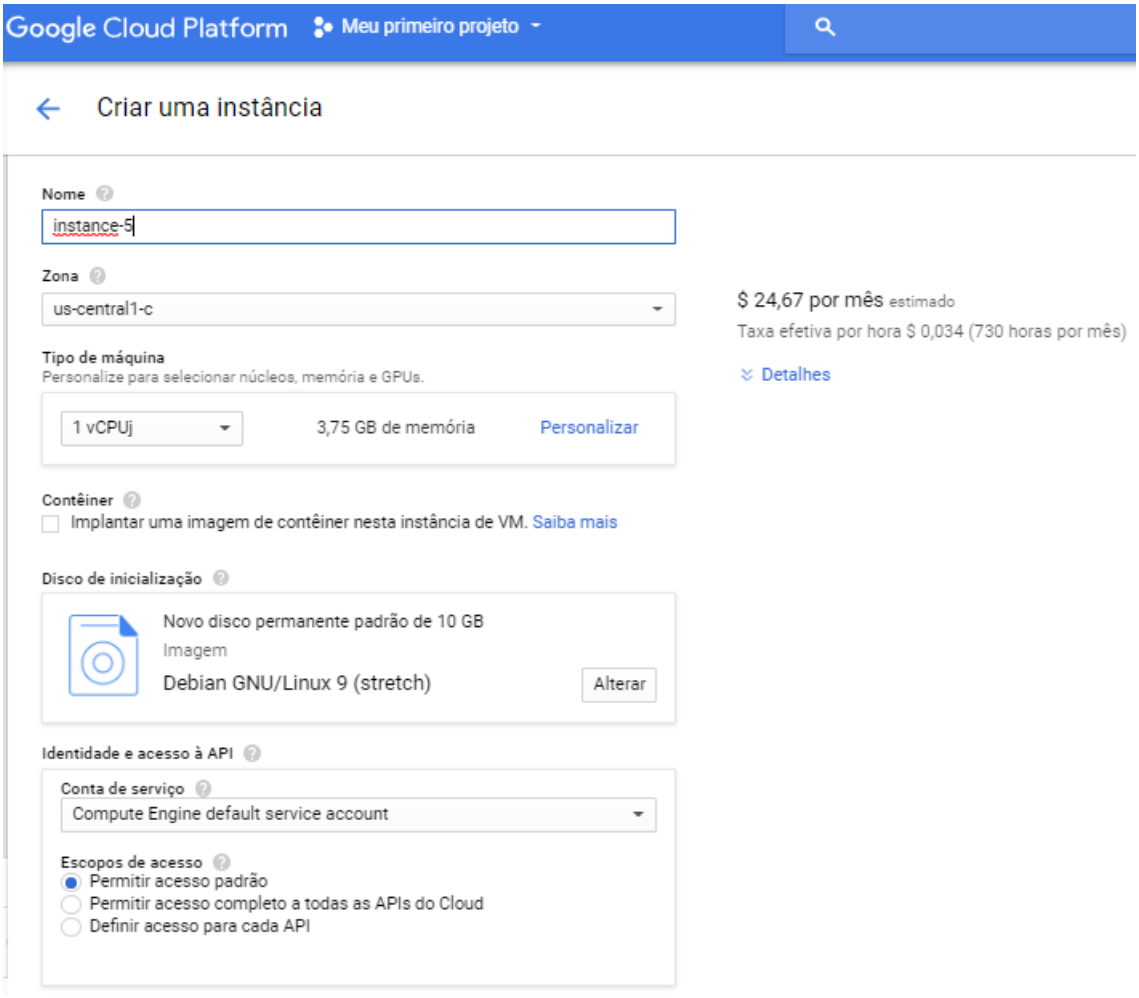


Figura 3.2: Criação e configuração de instância no provedor Google Cloud Platform.

3.1.2 Modelagem Conceitual da Proveniência de *Workflows* da Bioinformática Executados em Nuvem

A fim de descrever as questões sobre o uso de proveniência de dados da Bioinformática, em ambiente de nuvem computacional, foram definidos elementos e relações com base na modelagem do PROV-DM. Neste trabalho serão utilizados os elementos já definidos por de Paula (2012) como Projeto, Atividade, Agente e Entidade. Os principais requisitos do trabalho de de Paula (2012) foram a capacidade de representar a proveniência de um dado, descrevendo os processos e insumos utilizados em sua geração; e uma representação gráfica satisfatória, com diferentes símbolos para cada elemento, e relações suficientes para demonstrar a proveniência de forma objetiva. Assim, a utilização do modelo PROV-DM se mostrou simples e direta na questão de representar a proveniência de dados em projetos de bioinformática.

A Tabela 3.2 detalha as informações pertinentes a proveniência de dados de *workflow*

de Bioinformática. São detalhados os elementos já descritos na seção 2.4.2 e que são utilizados na modelagem conceitual.

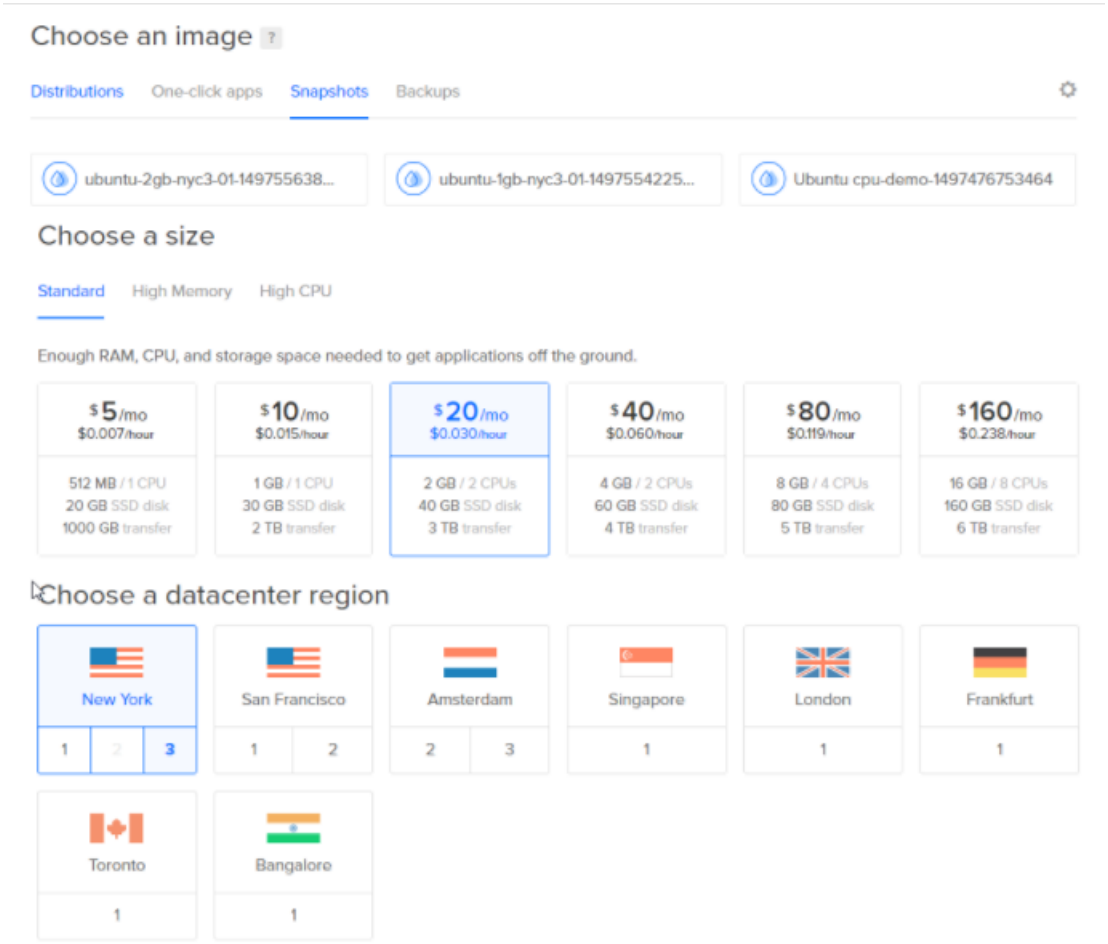


Figura 3.3: Criação e Configuração de Instância no Provedor DigitalOcean.

O modelo conceitual genérico para o armazenamento de proveniência de *workflow* em Bioinformática, em um ambiente de computação em nuvem, é apresentado por um Modelo de Dados Relacional na Figura 3.4. Note que, no diagrama, as tabelas na cor amarela são aquelas já definidas pelo trabalho de de Paula (2012) e as tabelas em azul tratam do ambiente de nuvem.

Um projeto de Bioinformática possui vários experimentos. Os experimentos podem ser compostos por várias atividades. As atividades por sua vez utilizam dados (arquivos) de entrada e também geram dados de saída. Cada atividade é executada por um agente responsável pela sua execução. As atividades são executadas em ambiente de nuvem computacional, ou seja, em uma máquina virtual instanciada no provedor. Várias máquinas podem ser instanciadas e utilizadas nos experimentos. A máquina está em uma localização (*datacenter*) e também pode fazer parte de um *cluster*. A atividade do *workflow*

Tabela 3.2: Elementos da Proveniência em Projetos da Bioinformática.

Projeto		Experimento	
Nome	Nome do projeto	Nome	Nome do experimento
Descrição	Descrição do provedor	Descrição	Descrição do experimento
Instituições Financiadoras	Instituições que financiaram o projeto	Local	Local de execução
Instituições Participantes	Instituições que participam do projeto	Data Inicial	Data inicial da execução
Coordenador	Nome do coordenador do projeto	Data Final	Data final da execução
Data Inicial	Data de início do projeto	Versão e Data	Número e data da versão gravada
Data Final	Data final do projeto	Anotações	Qualquer informação adicional sobre o projeto
Agente		Atividade	
Nome	Nome do usuário	Nome	Nome da atividade
Instituição	Instituição do usuário	Programa	Nome do programa
Cargo	Cargo ou função	Versão	Versão do programa
Anotações	Qualquer informação adicional sobre o agente	Comando	Linha de comando com os parâmetros utilizados
		Data inicial	Data e hora em que a atividade foi iniciada
		Data final	Data e hora em que a atividade foi concluída
		Anotações	Qualquer informação adicional sobre a entidade
Arquivo			
Nome		Nome do arquivo	
Descrição		Descrição do arquivo	
Localização		Localização do arquivo	
Anotação		Qualquer informação sobre o arquivo	
Tamanho		Tamanho do arquivo	
Arquivo		Arquivo físico	
Data de inserção		Data de inserção do arquivo	
Tipo		Tipo do arquivo	

é executada em uma máquina instanciada no ambiente de nuvem e utiliza e gera dados (arquivo).

Alguns elementos receberam atributos a mais relativos ao ambiente de execução. Por exemplo, a entidade Experimento possui atributo como custo total de execução ("custo_exec"), que informa o valor total da execução do experimento em determinada nuvem computacional. Esta informação advém do valor da máquina utilizada no provedor e do tempo de execução do *workflow*. Essas informações são obtidas por meio do relacionamento entre Atividades executadas no *workflow* e a Máquina do ambiente de nuvem onde as atividades foram executadas.

Este modelo conceitual foi utilizado como base para a construção dos modelos lógicos dos bancos de dados alvos deste trabalho. Os modelos são descritos na seção seguinte.

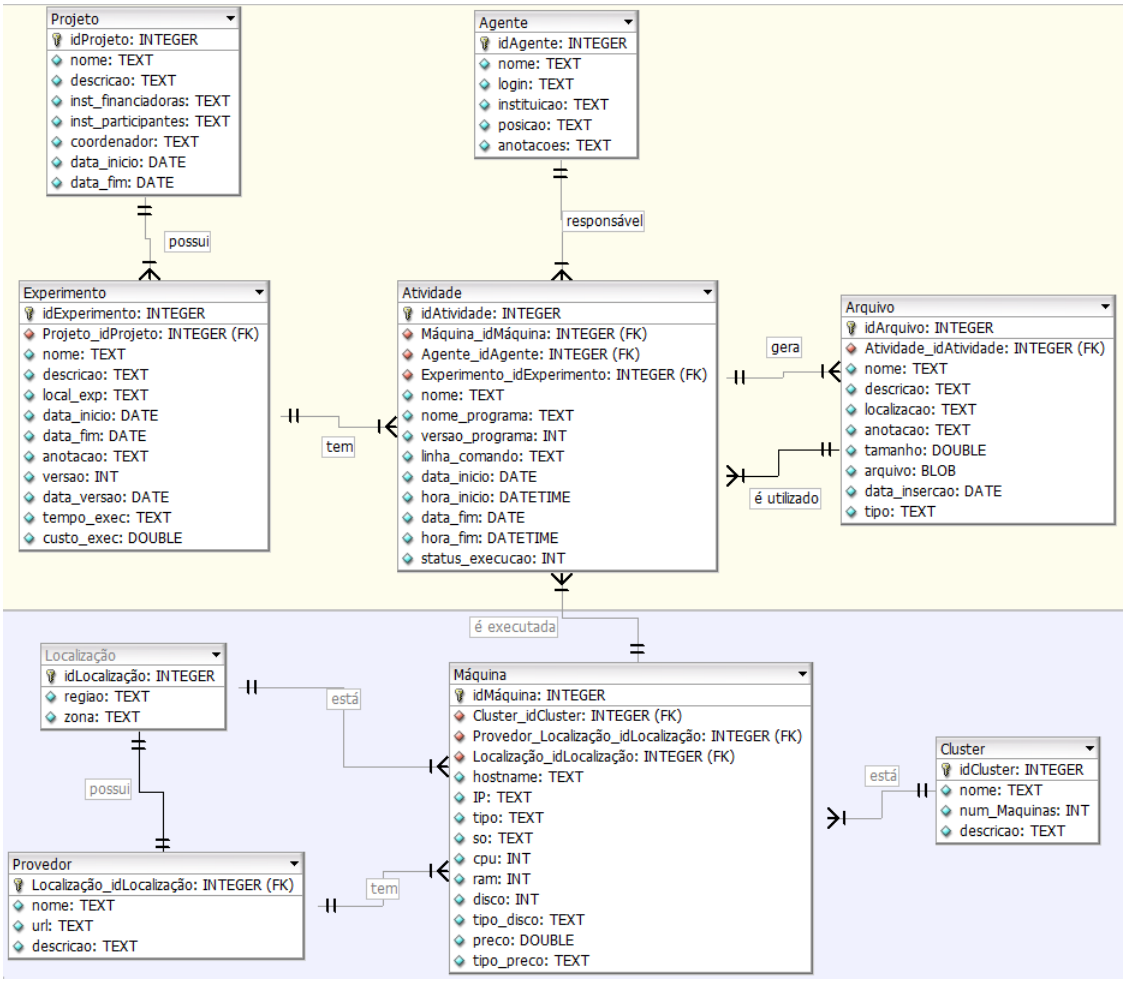


Figura 3.4: Modelo de Dados Relacional Genérico para a Proveniência de Dados em Projetos da Bioinformática em Nuvem.

3.2 Mapeamento das Modelagens

Uma das fases importantes em um projeto de banco de dados é a modelagem de dados, porque é através dela que é possível descrever e compreender as principais necessidades e características do projeto. Um entendimento claro do projeto é capaz de impedir erros de programação e de operação. Por isso, muitos consideram a fase de modelagem a parte mais importante em um projeto de banco de dados.

Neste contexto, para que a proveniência fosse capturada nos sistemas NoSQL escolhidos, um modelo lógico apropriado para cada arquitetura foi proposto, baseados no modelo

genérico apresentado na Seção 3.1.2. Os modelos lógicos propostos levaram em conta as características dos sistemas Cassandra (baseado em colunas), MongoDB (baseado em documentos) e OrientDB (baseado em grafo). As subseções seguintes descrevem os modelos propostos.

3.2.1 OrientDB

O OrientDB é um sistema híbrido, e implementa diferentes modelos de NoSQL, tais como documento, grafo, chave-valor e objeto. Nesta dissertação foi considerado o modelo de grafos do OrientDB. Segundo (ROBINSON *et al.*, 2013) as vantagens de se utilizar o modelo orientado a grafos são: alto desempenho independentemente do tamanho total do conjunto de dados e o modelo de grafos aproxima os domínios técnicos e de negócios facilitando a modelagem dos dados, e a capacidade de se alterar o esquema de dados incluindo novas entidades e relacionamentos, sem a necessidade de reestruturar o esquema de dados.

Nesse tipo de modelo os dados são representados por nós, arestas e propriedades. Os nós representam as entidades, as arestas representam as relações entre os nós e as propriedades apresentam características das entidades e dos relacionamentos. As arestas têm significância direcional que indicam como os nós se relacionam.

A Figura 3.5 monstra o modelo proposto em grafo para o armazenamento da proveniência no OrientDB. As entidades do modelo conceitual proposto foram mapeadas para nós do grafo, assim, as entidades projeto, experimento, atividade, arquivo, agente, máquina, *cluster*, provedor e localização transformaram-se em nós do grafo, e os relacionamentos entre essas entidades foram mapeados para arestas. Já a Figura 3.6 mostra um exemplo da criação de um vértice do tipo Atividade e a aresta E que marca o relacionamento com o vértice Experimento.

3.2.2 MongoDB

O MongoDB é um NoSQL baseado em documentos. Como já descrito anteriormente, os documentos são armazenados em coleções. Eles podem se relacionar de duas maneiras: por referência, quando existe uma referência ou *link* entre dois documentos; ou por meio de documentos embarcados, no qual os documentos são dispostos em campos ou em *arrays* de um documento.

Para a modelagem de documentos deste trabalho, as duas formas de relacionamento foram utilizadas, documentos embarcados para os elementos pertinentes aos diferentes experimentos científicos, e também documentos embarcados para os elementos pertinentes

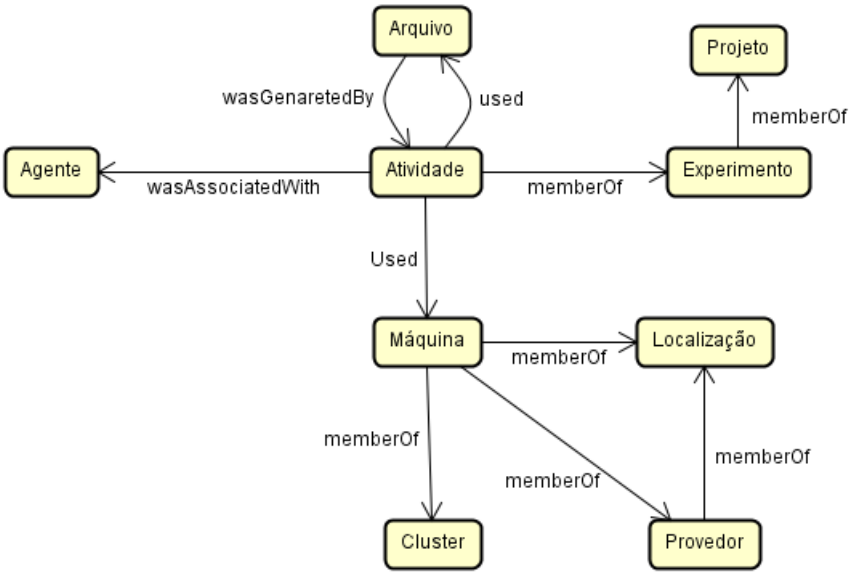


Figura 3.5: Modelagem Proposta para os Sistemas de Grafo.

```
orientdb> create class Atividade extends V

orientdb> create vertex atividade set id_Atividade = '1', nome_Ativ: 'At_Filt_Sickle',
nome_programa_Ativ: 'Sickle', versao_programa_Ativ: '1.33', linha_comando: 'sickle se
--fastq-file SRR5181508.fastq --qual-type sanger --output-file SRR5181508_FILTERED.fastq
-q 30 -l 25', data_inicio: '2018-01-02', hora_inicio: '15:17:37', data_fim: '2018-01-02',
hora_fim: '15:19:07', status_execucao: '2'
create record with RID #10:0

orientdb> create class memberOf extends E

orientdb> create edge memberOf from (select from atividade where id_Atividade = '1')
to (select from experimento where id_Exp = '1')
```

Figura 3.6: Criação de um vértice Atividade e a aresta E que a relaciona com Experimento no OrientDB.

aos diferentes ambientes computacionais, gerando coleções (*collections*) independentes que se relacionam por *links*.

Como mostra a Figura 3.7, o relacionamento na forma de documentos embarcados se mostra entre as entidades Projeto, Experimento, Atividade e Agente, assim como entre as entidades Provedor, Localização, *Cluster* e Máquina. Já a relação por referência está representada entre as entidades Atividade e Máquina, que nos diz que aquela atividade do *workflow* executou em uma determinada máquina na nuvem, assim como arquivos utilizados/gerados que estão armazenados naquela máquina.

A Figura 3.8 mostra o documento JSON criado referente a projeto, experimento e atividade. Como mostrado na modelagem, esses três elementos formam um documento

embarcado.

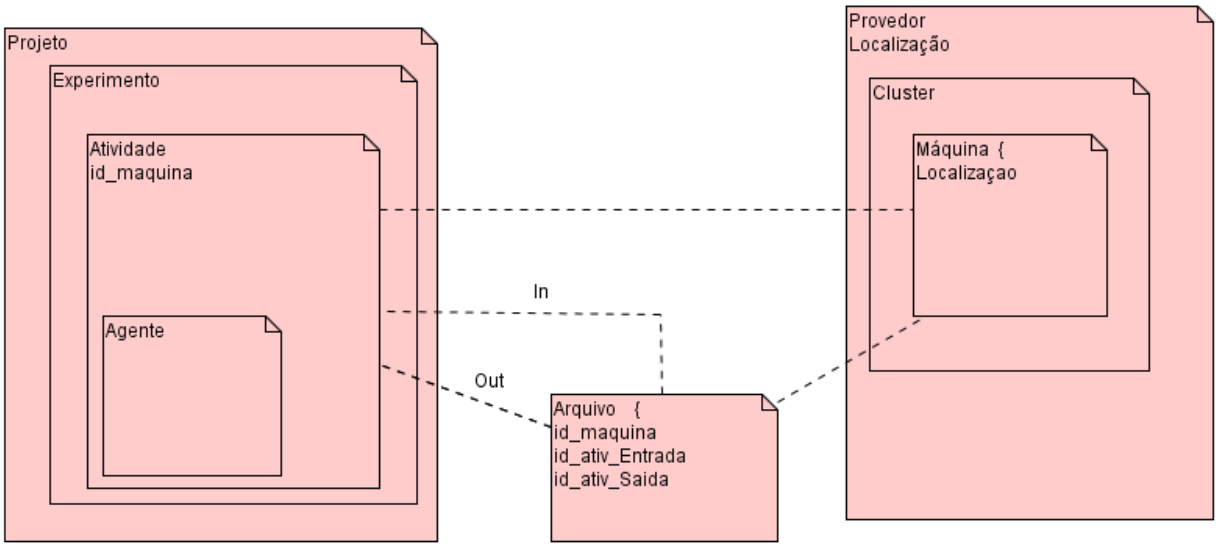


Figura 3.7: Modelagem Proposta para Sistemas Baseados em Documentos.

3.2.3 Cassandra

O Cassandra é um sistema gerenciador de banco de dados não relacional, baseado em colunas. Segundo Chebotko *et al.* (2015), o modelo de dados no Cassandra se atenta em como o dado será consultado pela aplicação, e busca preparar seu modelo de dados para atender as consultas com apenas uma tabela. Assim, no Cassandra, é importante analisar e implementar as consultas que serão realizadas pela aplicação no banco de dados a fim de gerar tabelas mais adequadas e eficientes. Isso porque o modelo de dados do Cassandra foi desenhado para obter o tempo mínimo de resposta, mesmo em operações complexas, sem *joins* ou agregações, presentes na linguagem SQL.

Sendo assim, analisando as informações que são necessárias para responder às questões dos biólogos, foi proposto um modelo orientado à consulta (CHEBOTKO *et al.*, 2015), baseando-se em consideração como os dados serão acessados. Para os testes desta dissertação, o modelo proposto abrange duas tabelas considerando as consultas pertinentes aos experimentos executados e ao ambiente de execução, porém, através do modelo conceitual proposto, é possível modelar tabelas com outras informações ou com mais informações de acordo com o que é necessário recuperar dos experimentos. A Figura 3.9 mostra o modelo desenvolvido para o Cassandra para esta dissertação.

Neste modelo proposto foram consideradas consultas pertinentes ao experimento executado, no qual a tabela ExpBioActivity foi implementada com os atributos de informa-

```
project_doc = {
"id": "1",
"nome": "Gerenciando dados de proveniencia em nuvem",
"descrição": "Gerenciando dados de proveniencia em nuvem",
"inst_financiadoras": "University of Brasilia",
"inst_participantes": "University of Brasilia",
"coordenador": "Maristela Holanda",
"data_inicial": "2018-01-02",
"data_final": "2018-04-20",
"experimento": {
               "id": "1",
                "nome": "RNA-seq de células endoteliais humanas",
                "descricao": "Workflow para mapear reads a uma referência",
                "local": "University of Brasília",
                "data_inicial": "2018-01-02",
                "data_final": "2018-01-02",
                "anotacoes": "Workflow de teste",
                "versao": "1",
                "data_versao": "2018-01-02",
                "tempo_exec": "19:59",
                "custo_execucao": "",
                "atividade": [{
                        "id_Atividade": "1",
                        "nome": "At_Filt_Sickle",
                        "nome_programa": "Sickle",
                        "versao_programa": "1.33",
                        "linha_comando": "sickle se --fastq-file SRR5181508.fastq --qual-type sanger
                                         --output-file SRR5181508_FILTERED.fastq -q 30 -l 25",
                        "data_inicio": "2018-01-02",
                        "hora_inicio": "15:17:37",
                        "data_fim": "2018-01-02",
                        "hora_fim": "15:19:07",
                        "status_execucao": "2"
                        "agent": {
                                "id": "1",
                                "name": "Ag_Fernanda",
                                "login": "fernanda.hondo",
                                "instituition": "University of Brasília",
                                "position": "Master's Degree student",
                                "annotation": ""
                         }
                         "files":[
                                  "3",
                                  "9"
                         ],
                         "maquina_execucao" : "1"
                 }]  } }
```

Figura 3.8: Exemplo da criação de um documento embarcado do tipo JSON referente a projeto, experimento e atividade.

ções do projeto, do experimento, bem como as atividades executadas no experimento, tais como nome do agente e arquivos de entrada e saída. Além disso, foi considerada para esta tabela como chave primária a composição de identificador do projeto e do experimento como chave de partição (K), e nome da ativadade, as datas inicial e final de atividade consideradas como *clustering key* (C). Já a tabela ExpBioCloud traz informações sobre o ambiente de execução dos experimentos, sendo considerada como chave primária o nome do provedor. A Figura 3.10 exemplifica a criação lógica da tabela ExpBioActivity no

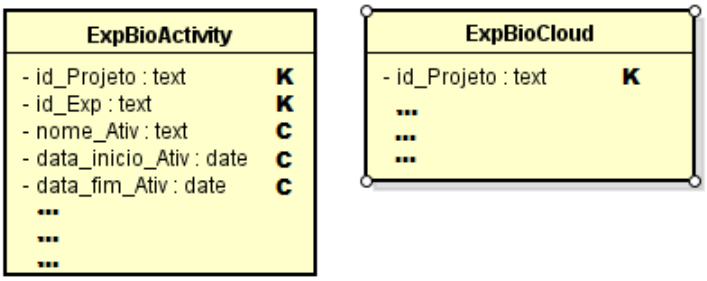


Figura 3.9: Modelagem Proposta para a Proveniência no Cassandra.

keyspace provenance criado.

```
cqlsh> CREATE KEYSPACE provenance
       WITH REPLICATION = {
       'class' : 'SimpleStrategy',
       'replication_factor' : 2
       };

cqlsh> CREATE TABLE provenance.ExpBioActivity (
        id_Projeto int,
        id_Projeto int,
        nome_Provedor text,
        nome_Ag text,
        nome_Ativ text,
        nome_Programa_Ativ text,
        versao_Programa_Ativ int,
        linha_Comando_Ativ text,
        data_Inicio_Ativ date,
        hora_Inicio_Ativ time,
        data_Fim_Ativ date,
        hora_Fim_Ativ time,
        nome_Arq_Entrada text,
        nome_Arq_Saida text,
        tipo_Arq_Entrada text,
        tipo_Arq_Saida text,
        blob_bin_Entrada blob,
        blob_bin_Saida blob,
   PRIMARY KEY ((id_Projeto, id_Exp), nome_Ativ, data_Inicio_Ativ, data_Fim_Ativ));
```

Figura 3.10: Criação de *keyspace* provenance e da tabela ExpBioActivity no Cassandra.

No Capítulo 4 serão apresentados os estudos de caso utilizados para a validação da modelagem proposta neste capítulo, os *workflows* utilizados, como o ambiente foi estruturado para os testes e os resultados.

Capítulo 4

Validação do Modelo e Análise dos Resultados

Este capítulo apresenta os estudos de caso para a validação da modelagem da proveniência em nuvem proposta no capítulo anterior. Para isto, este capítulo está dividido em seções, como segue: a Seção 4.1 faz a caracterização do ambiente de execução com a descrição dos *workflows* utilizados, e a aplicação do modelos nas nuvens públicas da Google Cloud e DigitalOcean. A Seção 4.3 faz uma análise da utilização do modelo proposto através de consultas realizadas aos NoSQL e também uma comparação entre os NoSQL utilizados.

4.1 Caracterização do Cenário

Transcritômica é o estudo do transcritoma (o conjunto de transcrições de RNA que são produzidas pelo genoma) em circunstâncias específicas ou em uma célula específica usando métodos de alto desempenho. A comparação de transcritomas permite identificar quais genes são diferencialmente expressos em tais circunstâncias ou tipos de células. *Workflows* científicos reais da Bioinformática que utilizam transcrições de RNA foram escolhidos para serem executados neste trabalho para a coleta e armazenamento da proveniência:

- *Workflow* 1: conjunto genômico da bactéria *Enterobacter kobei* (Anexo I);
- *Workflow* 2: análise quantitativa de RNA-seq de células endoteliais humanas (Anexo II).

4.1.1 *Workflow* 1: Conjunto genômico da bactéria *Enterobacter kobei*

Infecções urinárias são umas das doenças mais comuns em todo o mundo, sendo a maioria dos casos causada por essas espécies de bactérias. O DNA do *E. O kobei* foi sequenciado usando a plataforma HiSeq (Illumina) gerando 100 bp *paired-end reads*(JUDGE *et al.*, 2016). Este *workflow* é composto pelas fases de filtragem, montagem e análise. Inicialmente, é utilizado o kit de ferramentas SRA (INFORMATION, 2011) para baixar e converter os arquivos brutos. A Figura 4.1 ilustra de maneira resumida este *workflow* que é composto pelas seguintes fases:

- Filtragem: arquivos no formato FASTQ (ERR885455_1 e ERR885455_2) são dados de entrada da ferramenta Trimmomatic (BOLGER *et al.*, 2014) para filtragem e corte, que gera arquivos correspondentes de formato FASTQ filtrados.
- Montagem: é feita a montagem, utilizando a ferramenta Abyss (SIMPSON *et al.*, 2009) que recebe como dados de entrada os arquivos FASTQ filtrados e são gerados cinquenta arquivos de prefixo ERR885455_KMER_28;
- Análise: a ferramenta Quast (GUREVICH *et al.*, 2013) é utilizada para a análise estatística dos resultados, que são arquivos .pdf e .html gerados no diretório especificado.

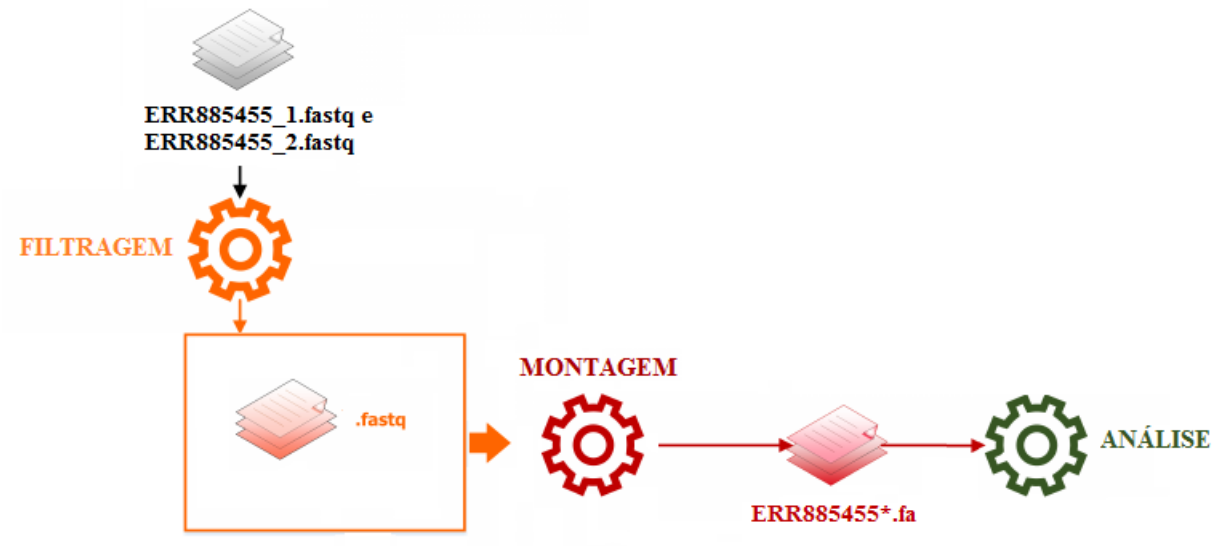


Figura 4.1: Fases do *workflow* 1 utilizado neste trabalho.

4.1.2 *Workflow* 2: RNA-seq de células endoteliais humanas

Este *workflow* consiste em mapear *reads* de DNA (cDNA) de células endoteliais microvasculares cardíacas primárias humanas e células endoteliais endocárdicas derivadas de hPSC (*Human Pluripotent Stem Cell*) para o cromossomo 22 do genoma humano (genoma de referência). Ele é composto por três fases: filtragem, mapeamento e análise. O arquivo de entrada é disponibilizado no formato .sra (*Sequence Read Archive*) com tamanho de 833.4 MB e somente depois de sua conversão, pela ferramenta sraatoolkit (INFORMATION, 2011) para o formato FASTQ, ele pode então ser utilizado. A Figura 4.2 ilustra este *workflow* e suas seguintes fases:

- Filtragem: é realizada a filtragem do arquivo de entrada contendo as *reads* de RNA-seq de células endoteliais cardíacas com a ferramenta Sickle(JOSHI *et al.*, 2011), em formato FASTQ, de acordo com um filtro estipulado na pesquisa;
- Mapeamento: é utilizada a ferramenta Hisat2 (KIM *et al.*, 2015) para realizar o alinhamento das *reads* filtradas na fase anterior com o genoma de referência humano (somente com o cromossomo 22). Esta fase gera um arquivo de saída no formato .SAM;
- Análise: é realizada a contagem. Um arquivo .BAM, gerado através da conversão e ordenação do arquivo de saída da fase de mapeamento, é utilizado como dado de entrada para a ferramenta Htseq (ANDERS *et al.*, 2015). É feito o processamento dos dados de RNA-Seq para análise de expressão diferencial contando a sobreposição de leituras com genes do genoma humano (*Homo_sapiens.GRCg38.88.gtf*).

4.1.3 Nuvem Computacional - Google Cloud Platform

Seguindo o fluxo de criação das instâncias no Google Cloud, a localização escolhida para as máquinas foi Leste dos EUA, Região - Carolina do Sul (*us-east1*) e Zona - *b*. Esta região foi escolhida devido ao custo mais baixo das instâncias. A Figura 4.3 mostra as instâncias criadas neste provedor. As instâncias foram criadas com as seguintes configurações:

- Uma máquina Linux- Ubuntu 14.04 com 2 núcleos de CPU, 2.5 GHz, Intel Xeon Family, 4 GB de RAM e 50GB de SSD para a execução do *workflow*;
- Três máquinas com CentOS 7 com 2 núcleo de CPU, 2.5 GHz, Intel Xeon Family, 4 GB de RAM e 30GB SSD para o armazenamento dos dados de proveniência.

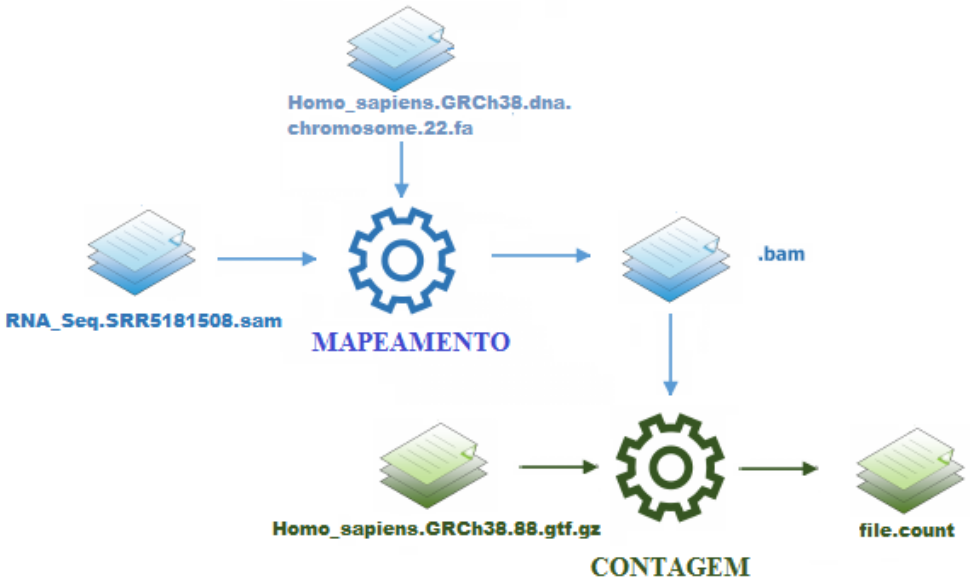


Figura 4.2: Fases do *workflow* 2 utilizado neste trabalho.

4.1.4 Nuvem Computacional - DigitalOcean

Seguindo os passos para criação de instâncias deste provedor, as máquinas foram criadas com as mesmas configurações das máquinas do Google Cloud, a localização escolhida foi a região Nova York, zona 2, por apresentar custos mais baixos de execução:

- Uma máquina Linux- Ubuntu 14.04 com 2 núcleo de CPU, 2.5 GHz, Intel Xeon Family, 4 GB de RAM e 30GB de SSD para a execução do *workflow*;
- Três máquinas com CentOS 7.4 com 2 núcleo de CPU, 2.5 GHz, Intel Xeon Family, 4 GB de RAM e 30GB SSD para o armazenamento dos dados de proveniência.

4.1.5 Estrutura do Ambiente para as Nuvens

Os dados brutos (arquivos de entrada) foram armazenados na mesma máquina em que os *workflows* foram executados. A Figura 4.4 mostra a estrutura do ambiente configurado para a execução deste projeto. Na instância com Ubuntu (Instance-4), envolvida na execução dos *workflows*, foram instaladas e configuradas as ferramentas de Bioinformática. Os três bancos de dados foram configurados para operar em *cluster* da seguinte maneira: para o Cassandra e o OrientDB foram utilizadas duas máquinas do *cluster*, Instance-1 e Instance-2, com fator de replicação (RF) 2, utilizando a estratégia *SimpleStrategy* (HEWITT, 2010) no Cassandra. Já para o MongoDB foram utilizados os componentes *Replica sets* e *shards*(CHODOROW, 2013) para configurar a mesma forma de trabalho em

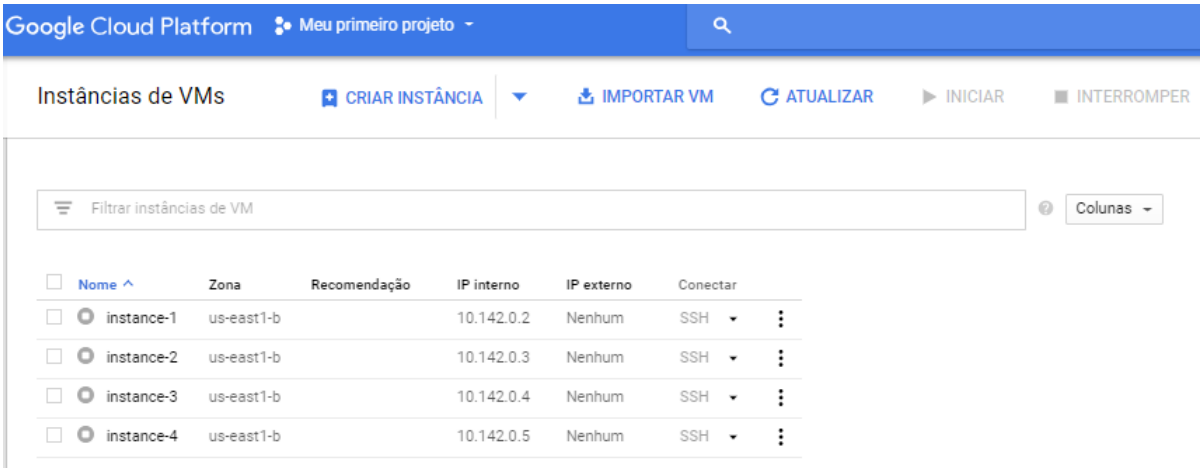


Figura 4.3: Instâncias criadas no provedor Google Cloud Platform.

cluster dos demais NoSQL, assim, foram utilizadas uma máquina para o nó primário (Instance-1) e duas para os nós secundários (*shards* - Instance-2 e Instance-3).

Para a validação dos modelos propostos foram feitos testes nas nuvens públicas destacadas nas subseções anteriores. Todo o ambiente de execução foi configurado da mesma forma, com as mesmas características para todos *workflows*. As versões dos NoSQL foram: Cassandra 3.11.2, MongoDB 2.6.5, OrientDB 2.2.33. Uma aplicação foi desenvolvida para realizar as chamadas das atividades de cada *workflow* e capturar os dados de proveniência e posteriormente os dados de proveniência foram armazenados nos três bancos de dados NoSQL de forma distribuída no *cluster*.

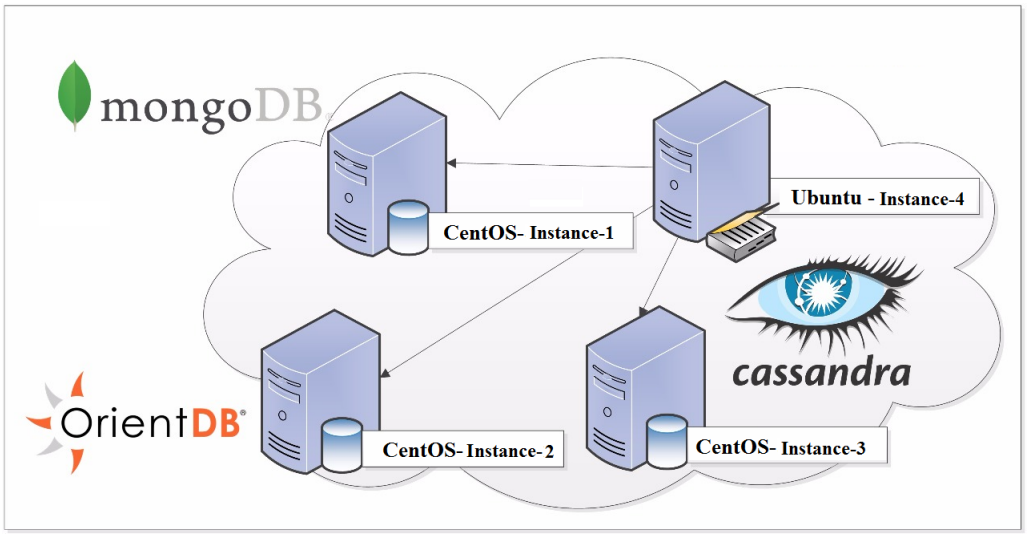


Figura 4.4: Ambiente utilizado na execução do projeto.

4.2 Resultados das Execuções dos *Workflows*

Os testes das execuções dos *workflows* para os ambientes de nuvem foram realizados da seguinte maneira: os *workflows* foram executados com a captura e armazenamento de dados de proveniência utilizando o modelo proposto para cada banco de dados NoSQL e seus tempos de execução total foram contados. As execuções para a captura e inserção dos dados de proveniência foram repetidos para verificar a consistência dos dados coletados e possíveis variações no desempenho.

Também foram gerados os grafos de proveniência de acordo com o modelo PROV-DM para cada *workflow* de acordo com as fases executadas dos *workflows*. Pra gerar os grafos, uma API Java, chamada Prefuse[1], foi utilizada.

4.2.1 Execução do *Workflow* 1

Como detalhado no sessão anterior, um mesmo ambiente de teste foi utilizado para a execução dos *workflows*. Foram realizadas três execuções para cada um dos bancos Cassandra, MongoDB e OrientDB em cada uma das nuvens, DigitalOcean e Google Cloud. O *workflow* é composto pelas fases de filtragem, montagem e análise.

O tempo médio de execução com a captura e armazenamento dos dados de proveniência para os bancos de dados utilizados na nuvem da DigitalOcean e Google Cloud foram, respectivamente, 1h40m11s e 20m06s.

Os grafos de proveniência foram gerados por meio da API Java Prefuse, a partir dos dados armazenados em cada um dos bancos de dados. A Figura 4.5 mostra o grafo de proveniência gerado para a fase de filtragem deste *workflow*, sendo o retângulo alaranjado a representação da atividade envolvida nesta fase (At_Filt_Trimmomatic), o agente responsável pela execução desta atividade (Ag_Fernanda), representado pelo pentágono cinza, e os dados brutos, representados pela elipse na cor azul, utilizados nessa atividade (ERR885455_1.fastq e ERR885455_2.fastq) e gerados por esta fase que são os arquivos ERR885455_1_FILTERED.fastq, ERR885455_2_FILTERED.fastq, ERR885455_1_UNPAIRED.fastq e ERR885455_2_UNPAIRED.fastq.

Já a Figura 4.6 mostra parte do grafo de proveniência produzido para a fase de montagem, os dados de entrada (ERR885455_1_FILTERED.fastq, ERR885455_2_FILTERED.fastq) utilizados pela atividade (At_Assembly_Abyss) para gerar as saídas de prefixo (ERR885455), sendo o agente executor dessa atividade o nó Ag_Fernanda.

[1] http://prefuse.org/

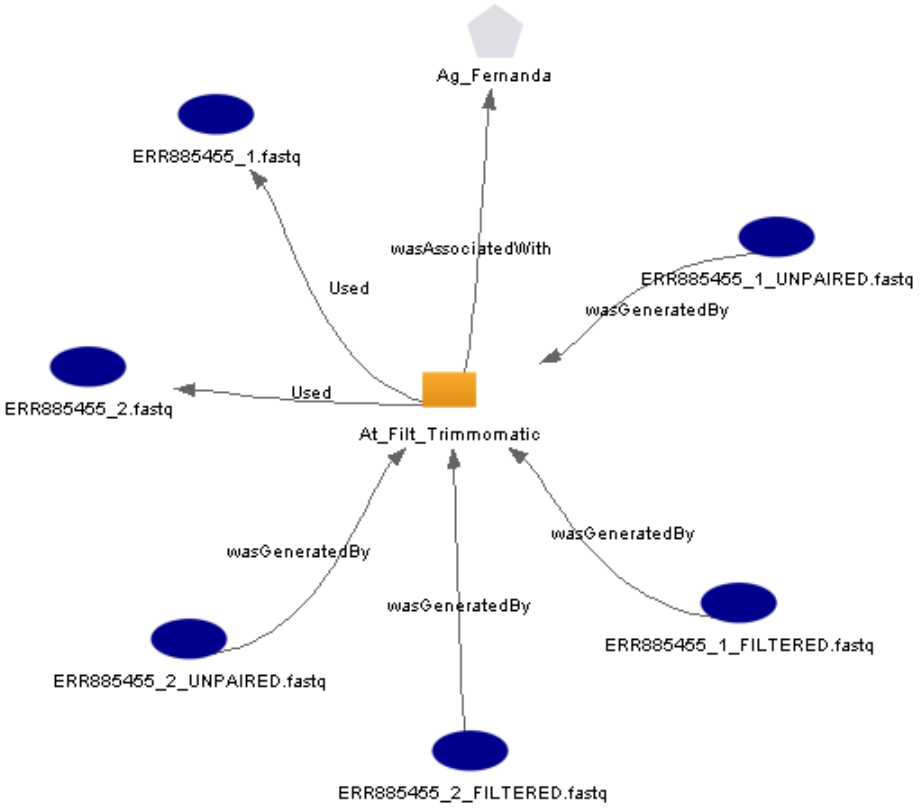


Figura 4.5: Grafo da fase de filtragem gerado para o *workflow* 1.

4.2.2 Execução do *Workflow* 2

Este *workflow* é composto pelas fases de filtragem, mapeamento e análise. Como detalhado na sessão anterior, este *workflow* recebe como dado de entrada um arquivo, contendo sequencias de RNA, que é filtrado, marcando a primeira atividade executada.

O tempo de execução do *workflow* 2 com a captura e armazenamento dos dados de proveniência para os bancos de dados utilizados na nuvem da DigitalOcean e Google Cloud foi de 1h27m29s e 18m07s, respectivamente.

O grafo de proveniência gerado para a fase de filtragem deste *workflow* pode ser visto na Figura 4.7, com o agente executor da atividade (Ag_Fernanda) representado na cor cinza, a atividade (At_Filt_Sickle), representada na cor alaranjada, o dado de entrada (SRR5181508.fastq) utilizado pela atividade e o dado de saída (SRR5181508_FILTERED.fastq) representados na cor azul.

A Figura 4.8 mostra o grafo gerado para a fase de mapeamento deste *workflow*, o agente executor(Ag_Fernanda) desta aticidade, a atividade (At_Map_Hisat2) executada nesta fase, os dados brutos (chromosome.22.hisat2.idx e SRR5181508_FILTERED.fastq) e dados gerados (SRR5181508.sam) por esta fase.

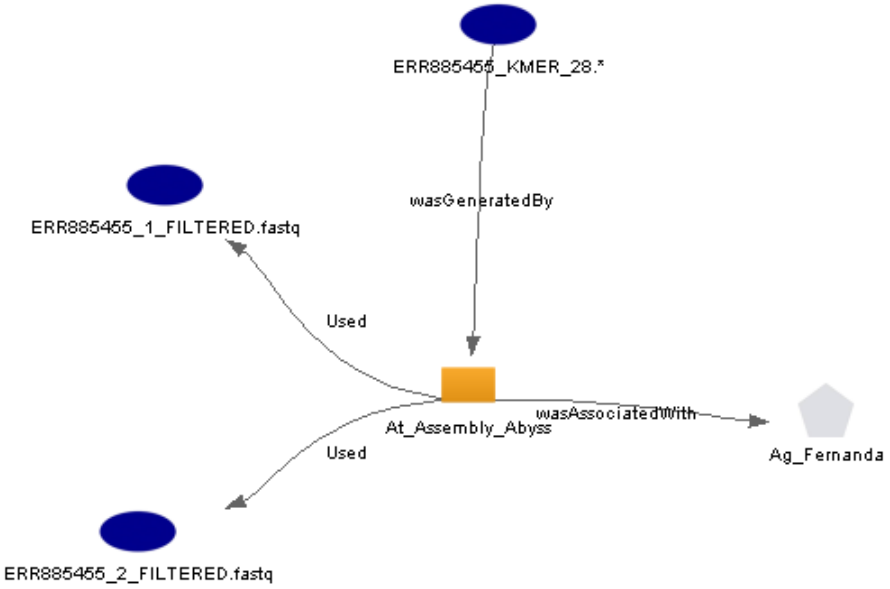


Figura 4.6: Grafo da fase de montagem gerado para o *workflow* 1.

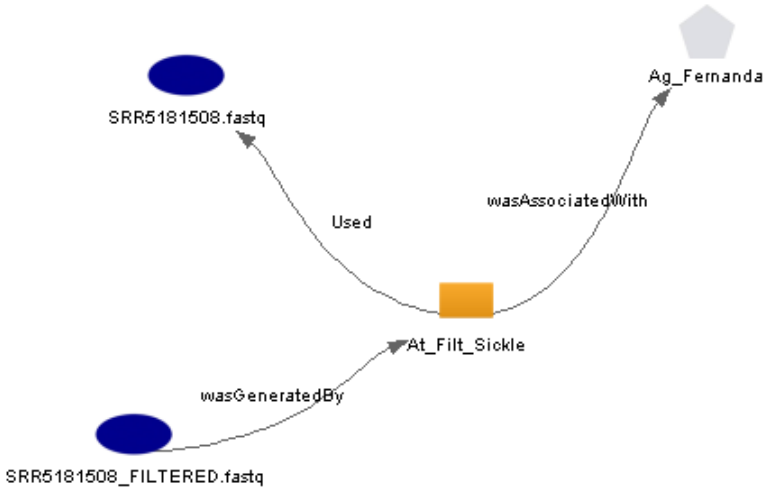


Figura 4.7: Grafo de proveniência gerado para a fase de filtragem do *workflow* 2.

4.3 Análise dos Resultados

Os *workflows* escolhidos para este trabalho foram executados em um mesmo ambiente, com as mesmas configurações. Para validar se as execuções e a captura dos dados de proveniência foram armazenadas corretamente, dentro da modelagem proposta por este trabalho, além das consultas realizadas para extrair informações para geração dos grafos de proveniênica, outras consultas foram elaboradas e implementadas para serem executadas nos bancos de dados.

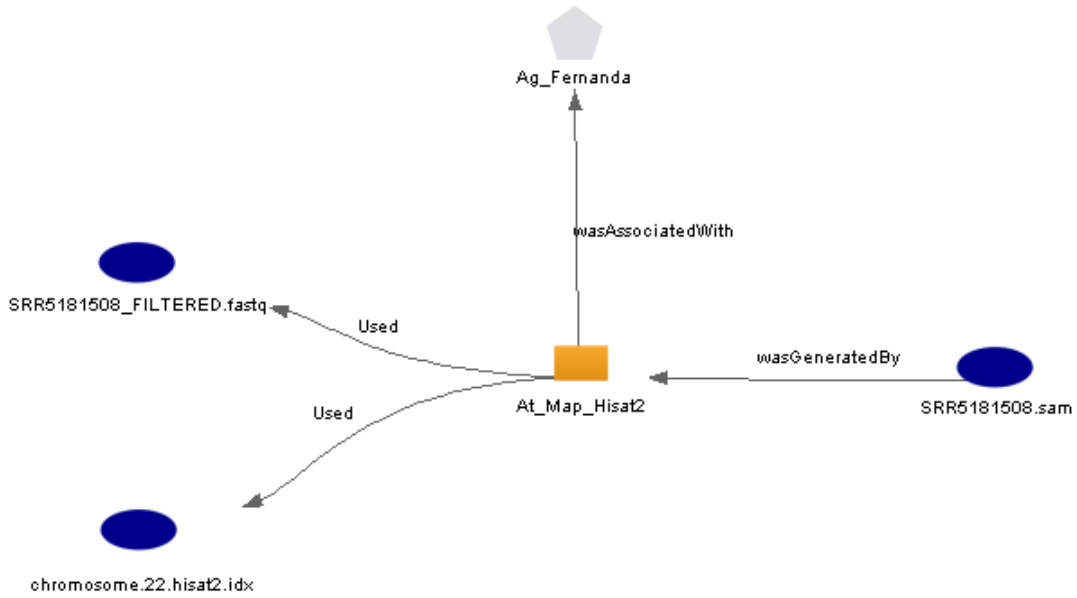


Figura 4.8: Grafo de proveniência gerado para a fase de mapeamento do *workflow* 2.

Consultas foram elaboradas para cada banco de dados utilizado neste trabalho, para que refletissem as principais questões sobre proveniência levantadas por um pesquisador do Departamento de Biologia da UnB quanto às execuções dos experimentos. Cada banco de dados tem sua própria linguagem de consulta e, em alguns casos, para obter as respostas às solicitações, foi necessário criar funções especiais, como obter o custo total do *workflow*. A exceção foi o caso do MongoDB, que tem suporte nativo para operadores aritméticos. A seguir, as consultas detalhadas.

4.3.1 Consulta 1: Atividades do *Workflow* 1

A primeira consulta foi feita para o *workflow*1 e leva em consideração as atividades que foram executadas durante o experimento científico, bem como as ferramentas utilizadas e suas versões: 'Quais as atividades, ferramentas e suas versões executadas no *workflow* 1 ?'

A Tabela 4.1 mostra as consultas realizadas para cada um dos bancos de dados, MongoDB, Cassandra e OrientDB utilizando suas próprias linguagens de consulta. Para cada uma delas, é necessário informar qual o é projeto (id_Projeto = 1), qual é o experimento em questão (experimento=1) e os dados a serem recuperados pelas consultas, sendo os nomes dos programas e suas versões utilizados na execução do *workflow* 1.

Tabela 4.1: *Query* 1

MongoDB	db.projeto.aggregate([{ $match : {$and : [{id : "1"}, { "experimento.id":"1" },]} }, { $unwind: "$experimento.atividade" }, { $group: { _id: { programa: '$experimento.atividade.programa_nome', versao:'$experimento.atividade.programa_versao' }}}])
Cassandra	SELECT nome_programa_Ativ, versao_programa_Ativ FROM provenance.ExpBioActivity WHERE id_Projeto =1 and id_Exp=1;
OrientDB	SELECT nome_programa_Ativ, versao_programa_Ativ FROM atividade GROUP BY nome_programa_Ativ WHERE id_Projeto=1 and id_Exp=1;

4.3.2 Consulta 2: Configuração do Ambiente

A segunda questão foi sobre a configuração do ambiente Google Cloud utilizado para execução dos experimentos científicos: 'Qual a configuração do ambiente Google Cloud (máquinas, processadores, memória , etc.)?'

A Tabela 4.2 detalha cada uma das consultas realizadas para cada um dos bancos de dados MongoDB, Cassandra e OrientDB. Para cada uma delas, é necessário informar qual o é o provedor em questão (nome_Provedor= Google Cloud) e os dados a serem recuperados pelas consultas, como as máquinas utilizadas, o *cluster* e a quantidade de máquinas envolvidas na execução.

Tabela 4.2: *Query* 2

MongoDB	db.provedor.find({nome_Provedor: "Google Cloud" , "cluster.nome_Cluster": "provBio"}, {_id: 0, "cluster.num_Maquinas_Cluster": 1, maquina: 1})
Cassandra	SELECT num_Maquinas_Cluster, tipo_Maq, so_Maq, cpu_Maq, ram_Maq, disco_Maq, tipo_disco_Maq, regiao_Maq, zona_Maq, preco_Maq, FROM ExpBioCloud WHERE nome_Provedor = "Google Cloud";
OrientDB	SELECT num_Maquinas_Cluster, tipo_Maq, so_Maq, cpu_Maq, ram_Maq, disco_Maq, tipo_disco_Maq, regiao_Maq, zona_Maq, preco_Maq, FROM maquina WHERE nome_Provedor = "Google Cloud";

4.3.3 Comparação entre as Nuvens Computacionais e os NoSQL

Apesar de não ser o foco do trabalho fazer análise de desempenho entre os diferentes bancos de dados NoSQL utilizados nos dois diferentes ambientes, os tempos de execução nas diferentes nuvens foram armazenados e são apresentados na Tabela 4.3.

Em geral, para executar um *workflow*, a nuvem DigitalOcean levou em média um tempo 4 vezes maior na execução de cada uma das atividades, tanto para o *workflow* 1 quanto para o *workflow* 2 comparado ao tempo das execuções na nuvem Google Cloud. A Tabela 4.3 refere-se aos tempos de execução dos *workflows* nas duas nuvens computacionais utilizadas, com o tempo de execução dos *workflows* sem considerar a coleta e armazenamento de dados de proveniência (SP) e os tempos de execução dos *workflows* com a captura e armazenamento dos dados de proveniência para cada NoSQL. É possível perceber que em ambos os ambientes analisados, o tempo gasto com as operações de coleta e inserção de proveniência é mínimo quando comparado ao tempo de execução do *workflow*. Como por exemplo, para o *workflow* 1 (W1) quando se utilizou o MongoDB na nuvem Google Cloud, o tempo total de execução gasto para inserir a proveniência levou cerca de sete segundos a mais que a execução sem a coleta e armazenamento da proveniência. Já para o *workflow* 2 (W2) na DigitalOcean a coleta e armazenamento da proveniência levou cerca de um minuto e vinte e oito segundos a mais. Vale ressaltar aqui que nos tempos das execuções com proveniência foi considerado somente o armazenamento dos dados de proveniência sem os arquivos brutos, logo, o tempo de armazenamento dos arquivos brutos foram desconsiderados dessa contagem.

Tabela 4.3: Relação entre o tempo de execução, captura e armazenamento de proveniência

	DigitalOcean				Google Cloud			
	SP	MongoDB	Cassandra	OrientDB	SP	MongoDB	Cassandra	OrientDB
W1	1h38m36s	1h40m11s	1h40m11s	1h40m11s	19m59s	20m06s	20m06s	20m06s
W2	1h26m01s	1h27m29s	1h27m29s	1h27m29s	18m03s	18m07s	18m07s	18m07s

4.4 Resultados em Publicações

O principal objetivo deste trabalho é fornecer uma modelagem de proveniência de dados para projetos de Bioinformática projetados para executar em nuvem. Desta maneira pode-se ressaltar as contribuições feitas em relação a extensão do modelo PROV-DM. As contribuições importantes relacionadas ao PROV-DM referem-se a adição das entidades relacionadas ao ambiente de nuvem: Provedor, Local, Cluster e Máquina. Essas entidades permitem que os biólogos consigam obter informações importantes sobre o ambiente de

execução do experimento facilitando o seu gerenciamento e reprodução. A partir deste trabalho, foi elaborado o seguinte artigo submetido para revista:

- *Bioinformatics Workflows with NoSQL Database in Cloud Computing*, artigo submetido para o *Evolutionary Bioinformatics (2018)*.

E também foram elaborados dois artigos publicados nas seguintes conferências:

- Uso de Bancos de Dados NoSQL para Gerenciamento de Dados em *Workflow* de Bioinformática, artigo aceito para o *Workshop Databases meet Bioinformatics* do XXXII Simpósio Brasileiro de Banco de Dados (SBBD 2017), realizado em Uberlândia/MG em 2017;
- *Data Provenance Management for Bioinformatics Workflows Using NoSQL Databases in Cloud Computing Environment*, artigo aceito para o *The 4th International Workshop on High Performance Computing on Bioinformatics* do IEEE International Conference on Bioinformatics and Biomedicine (BIBM 2017) realizado em Kansas City, Estados Unidos;

Anteriormente, como início deste projeto de pesquisa, foram realizadas contribuições para o artigo publicado em revista:

- *Evaluating the Cassandra NoSQL Database Approach for Genomic Data Persistency*, artigo publicado no *International Journal of Genomics (2015)* Volume 2015.

E também foram elaborados os artigos publicados nas conferências:

- *A Study of Genomic Data Provenance in NoSQL Document-Oriented Database Systems*, artigo aceito para o *Workshop on High Performance Computing on Bioinformatics* do IEEE International Conference on Bioinformatics and Biomedicine (BIBM 2015) realizado em Washington D.C., Estados Unidos;
- *Data Modeling for NoSQL Document-Oriented Databases*, artigo aceito para o *2nd Annual International Symposium on Information Management and Big Data*, realizado em Cusco, Peru em 2015;

Capítulo 5

Conclusões e Trabalhos Futuros

Este trabalho apresentou uma modelagem de proveniência de dados de *workflows* científicos, especialmente em Bioinformática, com o uso de diferentes NoSQL para a execução dos *workflows* em nuvens computacionais e armazenamento dos dados, e utilização do modelo PROV-DM para a modelagem da proveniência. Os NoSQL escolhidos para este trabalho foram o Cassandra, o MongoDB e o OrientDB, que são sistemas consideravelmente influentes dentre os existentes, e diferentes modelos lógicos, das diferentes famílias de bancos de dados NoSQL foram implementados para o gerenciamento dos dados de proveniência e o armazenamento de dados biológicos em nuvens computacionais.

Para que cada um dos bancos de dados escolhidos pudesse armazenar os dados de proveniência com foco na melhor prática da utilização do PROV-DM, foi proposto uma modelagem conceitual genérica inicial, sendo posteriormente mapeado de acordo com cada uma das famílias dos bancos NoSQL utilizados neste trabalho.

Ao longo das execuções realizadas, foi possível observar que, o banco de dados OrientDB apresentou facilidade quanto a modelagem e armazenamento da proveniência nos moldes do PROV-DM, geração e visualização nativa de grafos, sendo possível navegar com facilidade pelo grafo, analisando as entidades e atividades envolvidas na geração dos dados e também as informações pertinentes a cada um dos nós do grafo como: a máquina em que a atividade executou e/ou que o dado está armazenado.

A partir da análise dos tempos de execução, inserção de dados de proveniência, pode-se observar que os três sistemas apresentaram desempenho satisfatório para o objetivo deste trabalho. Diante dos resultados obtidos, pode-se concluir que a modelagem proposta para cada um dos NoSQL para a captura e armazenamento dos dados de proveniência não causa impactos no tempo total da execução dos *workflows*.

O provedor de nuvem, Google Cloud, destacou-se pelo fato da facilidade e intuitividade de configuração de todo ambiente, a nível de usuários com pouca experiência em infraes-

trutura, o que é o caso dos pesquisadores e biólogos que trabalham com os experimentos alvos deste trabalho, e também pelo tempo gasto na execução dos *workflows*.

Assim, de forma geral, a modelagem proposta nesta dissertação mostrou-se viável, tendo em vista que explica de forma eficiente os requisitos de reprodutibilidade do experimento na nuvem, incluindo seus custos, tempos de execução e também não causa impactos de desempenho nas execuções dos *workflows*. Também é possível considerar que a modelagem proposta neste trabalho é genérica o sufuciente para ser adaptada a outros *workflows* científicos de outras áreas, não somente para a Bioinformática.

O uso de bancos de dados NoSQL para gerenciar o armazenamento da proveniência dos dados nos moldes do PROV-DM apresentou bons resultados. Desta forma, este trabalho aponta para melhorias da reprodutibilidade dos experimentos baseados em *workflows* de Bioinfomática.

Porém, novos experimentos precisam ser realizados expandindo a diversidade de *workflows*, considerando que este trabalho utilizou somente dois tipos de *workflows* dos diversos existentes na Bioinformática. É válido também, para otimização do trabalho dos pesquisadores, a montagem e configuração do ambiente de execução dos *workflows* de forma automatizada, através de tecnologias como Containers que trazem esta vantagem a experimentos.

Outro trabalho futuro a ser explorado é o uso de replicação dos bancos de dados para analisar e verificar o comportamento em um ambiente de processamento distribuído, aumentando o fator de replicação. Também considera-se como um trabalho futuro, testes em outros ambientes de nuvem, diferente dos utilizados nesta dissertação.

Referências Bibliográficas

ABRAMOVA, Veronika *et al.* Nosql databases: Mongodb vs cassandra. In: *Proceedings of the International C* Conference on Computer Science and Software Engineering*. New York, NY, USA: ACM, 2013. (C3S2E '13), p. 14–22. ISBN 978-1-4503-1976-8. 19

ALMEIDA, Rodrigo Pinheiro. *Proveniência de dados em workflow de Bioinformática utilizando banco de dados baseado em grafo*. Dissertação (Mestrado) — Universidade de Brasília, Departamento de Ciência de Computação, 2015. 21, 29

ANDERS, Simon *et al.* Htseq—a python framework to work with high-throughput sequencing data. *Bioinformatics*, Oxford University Press, v. 31, n. 2, p. 166–169, 2015. 45

ANICETO, Rodrigo *et al.* Evaluating the cassandra nosql database approach for genomic data persistency. *International journal of genomics*, Hindawi Publishing Corporation, v. 2015, 2015. 19

ARMBRUST, Michael *et al. Above the clouds: A berkeley view of cloud computing*. [S.l.], 2009. 11

ASSIS, Vanessa Marques de. *ProvDoop: captura, armazenamento e disponibilização de dados de proveniência em tempo de execução de sistemas sobre Hadoop*. Dissertação (Mestrado) — Universidade Federal do Rio de Janeiro, 2016. x, 10, 28, 29

BLEIDORN, Christoph. Assembly and data quality. In: *Phylogenomics*. [S.l.]: Springer, 2017. p. 81–103. 8

BOEKEL, Jorrit *et al.* Multi-omic data analysis using galaxy. *Nature biotechnology*, Nature Research, v. 33, n. 2, p. 137–139, 2015. 7

BOICEA, Alexandru *et al.* Mongodb vs oracle–database comparison. In: IEEE. *2012 Third International Conference on Emerging Intelligent Data and Web Technologies*. [S.l.], 2012. p. 330–335. 19

BOLGER, Anthony M *et al.* Trimmomatic: a flexible trimmer for illumina sequence data. *Bioinformatics*, Oxford University Press, v. 30, n. 15, p. 2114–2120, 2014. 44

BREWER, Eric. Cap twelve years later: How the"rules"have changed. *Computer*, IEEE, v. 45, n. 2, p. 23–29, 2012. 15

BUNEMAN, Peter *et al.* Why and where: A characterization of data provenance. In: BUSSCHE, Jan Van den *et al.* (Ed.). *Database Theory — ICDT 2001: 8th International Conference London, UK, January 4–6, 2001 Proceedings*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001. p. 316–330. ISBN 978-3-540-44503-6. 1, 21

BUYYA, Rajkumar *et al.* Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility. *Future Generation computer systems*, Elsevier, v. 25, n. 6, p. 599–616, 2009. 11

CATTELL, Rick. Scalable sql and nosql data stores. *Acm Sigmod Record*, ACM, v. 39, n. 4, p. 12–27, 2011. 14

CHEBOTKO, Artem *et al.* A big data modeling methodology for apache cassandra. In: IEEE. *Big Data (BigData Congress), 2015 IEEE International Congress on*. [S.l.], 2015. p. 238–245. 18, 40

CHODOROW, Kristina. *MongoDB: The Definitive Guide, 2nd Edition - Powerful and Scalable Data Storage*. [S.l.]: O'Reilly Media, 2013. 46

CORBELLINI, Alejandro *et al.* Persisting big-data: The nosql landscape. *Information Systems*, Elsevier, v. 63, p. 1–23, 2017. 15

CRUZ, Sergio Manuel Serra da *et al.* Collecting cloud provenance metadata with Matriohska: A case study with genomic workflows. In: *Proceedings of the 29th Annual ACM Symposium on Applied Computing*. New York, NY, USA: ACM, 2014. (SAC '14), p. 351–356. ISBN 978-1-4503-2469-4. 28, 29

DE PAULA, Renato. *Proveniência de dados em workflows de bioinformática*. Dissertação (Mestrado) — Universidade de Brasília Departamento de Ciência de Computação, 2012. 1, 9, 21, 25, 29, 34, 35

DEELMAN, Ewa *et al.* Workflows and e-Science: An overview of workflow system features and capabilities. *Future Generation Computer Systems*, v. 25, n. 5, p. 528 – 540, 2009. ISSN 0167-739X. Disponível em: <//www.sciencedirect.com/science/article/pii/S0167739X08000861>. 6

FERREIRA, Guilherme R. *et al.* Uso de sgbds nosql na gerência da proveniência distribuída em workflows científicos. *XXIX Simpósio Brasileiro de Bancos de Dados*, 2014. 26, 28, 29

FOSTER, Ian *et al.* Cloud computing and grid computing 360-degree compared. In: IEEE. *Grid Computing Environments Workshop, 2008. GCE'08*. [S.l.], 2008. p. 1–10. 11, 12

GOBLE, Carole. Position statement: Musings on provenance, workflow and (semantic web) annotations for bioinformatics. In: *Workshop on Data Derivation and Provenance, Chicago*. [S.l.: s.n.], 2002. v. 3. 1, 21

GOOGLE. *Google Cloud Platform Documentation - Overview*. 2017. Disponível em: <https://cloud.google.com/docs/overview/?hl=pt-br>. 32

GUIMARAES, Valeria *et al.* A study of genomic data provenance in nosql document-oriented database systems. In: IEEE. *Bioinformatics and Biomedicine (BIBM), 2015 IEEE International Conference on*. [S.l.], 2015. p. 1525–1531. 19

GUIMARÃES, Milene Pereira *et al. Proveniência de dados na área de Bioinformática*. Dissertação (Mestrado) — Universidade de Caxias do Sul, 2009. 22

GUO, Minzhe *et al.* Hands-on labs for learning mobile and nosql database security. In: IEEE. *Computer Software and Applications Conference (COMPSAC), 2016 IEEE 40th Annual*. [S.l.], 2016. v. 2, p. 606–607. 18

GUREVICH, Alexey *et al.* Quast: quality assessment tool for genome assemblies. *Bioinformatics*, Oxford University Press, v. 29, n. 8, p. 1072–1075, 2013. 44

HAAS, Brian J *et al.* De novo transcript sequence reconstruction from rna-seq using the trinity platform for reference generation and analysis. *Nature protocols*, Nature Publishing Group, v. 8, n. 8, p. 1494–1512, 2013. 8

HEWITT, Eben. *Cassandra: The Definitive Guide*. [S.l.]: O'Reilly Media, 2010. 18, 19, 46

HOLLINGSWORTH, D. *Workflow Management Coalition - The Workflow Reference Model*. [S.l.], 1995. 1

HOLT, Victoria *et al.* The usage of best practices and procedures in the database community. *Information Systems*, Elsevier, v. 49, p. 163–181, 2015. 18

HUACARPUMA, Ruben Cruz. *Modelo de dados para um Pipeline de seqüenciamento de alto desempenho transcritômico*. Dissertação (Mestrado) — Universidade de Brasília, Departamento de Ciência de Computação, 2012. 8

INFORMATION, SRA Toolkit National Center for Biotechnology. *Sequence Read Archive Submissions Staff. Using the SRA Toolkit to convert .sra files into other formats. In: SRA Knowledge Base*. 2011. [Online; acessado 19-Dez-2016]. Disponível em: <https://www.ncbi.nlm.nih.gov/books/NBK158900/>. 44, 45

JOSHI, NA *et al. Sickle: A sliding-window, adaptive, quality-based trimming tool for FastQ files (Version 1.33)[Software]*. 2011. 45

JUDGE, Kim *et al.* Comparison of bacterial genome assembly software for minion data and their applicability to medical microbiology. *Microbial genomics*, Microbiology Society, v. 2, n. 9, 2016. 44

KIM, Daehwan *et al.* Hisat: a fast spliced aligner with low memory requirements. *Nature methods*, Nature Publishing Group, v. 12, n. 4, p. 357, 2015. 45

KOHL, Michael *et al.* A practical data processing workflow for multi-omics projects. *Biochimica et Biophysica Acta (BBA)-Proteins and Proteomics*, Elsevier, v. 1844, n. 1, p. 52–62, 2014. 7

LAKSHMAN, Avinash *et al.* Cassandra: a decentralized structured storage system. *ACM SIGOPS Operating Systems Review*, ACM, v. 44, n. 2, p. 35–40, 2010. 18, 19

LIMA, Iasmini Virgínia O. *Replicação de Dados em Workflows de Bioinformática usando os Bancos de Dados NoSQL*. Dissertação (Mestrado) — Universidade de Brasília, 2016. 19

LOEFFLER, Bill. Cloud computing: what is infrastructure as a service. *TechNet Magazine*, v. 10, 2011. 12

MARINHO, A. *et al.* A strategy for provenance gathering in distributed scientific workflows. In: *2009 Congress on Services - I*. [S.l.: s.n.], 2009. p. 344–347. ISSN 2378-3818. 2, 21, 26

MATTOS, Amanda *et al. Gerência de Workflows Científicos: uma análise crítica no contexto da bioinformática*. [S.l.], 2008. 6, 25

MATTOSO, Marta *et al.* Gerenciando experimentos científicos em larga escala. *SBCSEMICH*, 2008. 6, 26

MATTOSO, Marta *et al.* Towards supporting the life cycle of large scale scientific experiments. *International Journal of Business Process Integration and Management*, Inderscience Publishers, v. 5, n. 1, p. 79–92, 2010. 10

MELL, Peter *et al.* The nist definition of cloud computing (draft). *NIST Special Publication*, v. 800, p. 145, 2011. 12

MOHAMED, Mohamed A *et al.* Relational vs. nosql databases: A survey. *International Journal of Computer and Information Technology*, v. 3, n. 03, p. 598–601, 2014. 14

MOTA, Evandro Roberto. *Bancos de dados geográficos : uma abordagem orientada a grafos*. 2016. Trabalho de Conclusão de Curso - Graduação - Licenciatura. 20

MUNISWAMY-REDDY, Kiran-Kumar *et al.* Provenance as first class cloud data. *ACM SIGOPS Operating Systems Review*, ACM, v. 43, n. 4, p. 11–16, 2010. 27, 29

OCAÑA, Kary ACS *et al.* Sciphy: a cloud-based workflow for phylogenetic analysis of drug targets in protozoan genomes. In: SPRINGER. *Brazilian Symposium on Bioinformatics*. [S.l.], 2011. p. 66–70. 28

OLIVEIRA, Ary HM de *et al.* Clouds and reproducibility: A way to go to scientific experiments? In: *Cloud Computing*. [S.l.]: Springer, 2017. p. 127–151. 28

OLIVEIRA, Daniel de *et al.* Scicumulus: A lightweight cloud middleware to explore many task computing paradigm in scientific workflows. In: IEEE. *Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on*. [S.l.], 2010. p. 378–385. 28

PAULINO, Carlos *et al.* Captura de metadados de proveniencia para workflows científicos em nuvens computacionais. *Anais Do XXV Simpósio Brasileiro de Banco de Dados*, 2010. 2

POURABBAS, Elaheh. *Geographical Information Systems:Trends and Technologies*. [S.l.: s.n.], 2014. ISBN 9781466596931. 15

PROSDOCIMI, Francisco *et al.* Bioinformática: manual do usuário. *Biotecnologia Ciência & Desenvolvimento*, v. 29, p. 12–25, 2002. 1

ROBINSON, Ian *et al. Graph databases*. [S.l.]: "O'Reilly Media, Inc.", 2013. 38

SALDANHA, Hugo V. *Bionimbus: uma arquitetura de federação de nuvens computacionais híbrida para a execução de workflows de bioinformática*. Dissertação (Mestrado) — Universidade de Brasília, Departamento de Ciência de Computação, 2012. x, 8, 11, 13, 14

SANTOS, Edimar *et al.* Distribuição de bases de dados de proveniência na nuvem. *Simpósio Brasileiro de Banco de Dados*, 2013. 28, 29

SIMPSON, Jared T *et al.* Abyss: a parallel assembler for short read sequence data. *Genome research*, Cold Spring Harbor Lab, v. 19, n. 6, p. 1117–1123, 2009. 44

SINGH, Kuldeep. *Survey of NoSQL Database Engines for Big Data*. 83 p. Dissertação (Mestrado) — Aalto University, School of Science, 2015. 19

TESORIERO, Claudio. *Getting Started with OrientDB*. [S.l.]: Packt Publishing, 2013. 20

VAQUERO, Luis M *et al.* A break in the clouds: towards a cloud definition. *ACM SIGCOMM Computer Communication Review*, ACM, v. 39, n. 1, p. 50–55, 2008. 11

W3C. *PROV-DM: The PROV Data Model*. 2018. [Online; acessado 04-Abril-2018]. Disponível em: <https://www.w3.org/TR/prov-dm/>. x, 22, 23, 24, 28

WOLSTENCROFT, Katherine *et al.* The taverna workflow suite: designing and executing workflows of web services on the desktop, web or in the cloud. *Nucleic acids research*, Oxford University Press, v. 41, n. W1, p. W557–W561, 2013. 7

ZERBINO, Daniel R *et al.* Velvet: algorithms for de novo short read assembly using de bruijn graphs. *Genome research*, Cold Spring Harbor Lab, v. 18, n. 5, p. 821–829, 2008. 8

ZHANG, Olive Qing *et al.* How to track your data: The case for cloud computing provenance. In: IEEE. *Cloud Computing Technology and Science (CloudCom), 2011 IEEE Third International Conference on*. [S.l.], 2011. p. 446–453. 27, 29

Anexo I

Workflow 1: Conjunto genômico da bactéria *Enterobacter kobei*

I.1 *Workflow* de montagem *de novo*

Illumina HiSeq 2000 paired end sequences available in European Nucleotide Archive (ENA) under accession number ERR885455

```
# download raw files (reads)
$ wget ftp://ftp.sra.ebi.ac.uk/vol1/fastq/ERR885/ERR885455/
   ERR885455_1.fastq.gz
$ wget ftp://ftp.sra.ebi.ac.uk/vol1/fastq/ERR885/ERR885455/
   ERR885455_2.fastq.gz

# Filtering with Trimmomatic

$ java -jar /opt/Trimmomatic-0.36/trimmomatic-0.36.jar PE -phred33
   ERR885455_1.fastq.gz ERR885455_2.fastq.gz ERR885455_1_FILTERED.
   fastq ERR885455_1_UNPAIRED.fastq ERR885455_2_FILTERED.fastq
   ERR885455_2_UNPAIRED.fastq ILLUMINACLIP:/opt/Trimmomatic-0.36/
   adapters/TruSeq3-PE.fa:2:30:10 LEADING:3 TRAILING:3 SLIDINGWINDOW
   :4:15 MINLEN:36 -validatePairs

# de novo assembly with ABySS
# abyss-pe for paired-end reads
# k        size of the kmer
# np       number of processors (depends on mpi modules installed)
# in       input file for forward/reverse trimmed reads
# name     output file prefix
```

```
$ abyss-pe in='ERR885455_1_FILTERED.fastq ERR885455_2_FILTERED.fastq'
    k=28 name=ERR885455_KMER_28 np=4


# Statistical analisys with QUAST
$ python /opt/quast/quast.py -o ERR885455_stats ERR885455*.fa
```

In the analisys phase, Quast generates visual reports for the assembly as can be seen in Figure I.1.

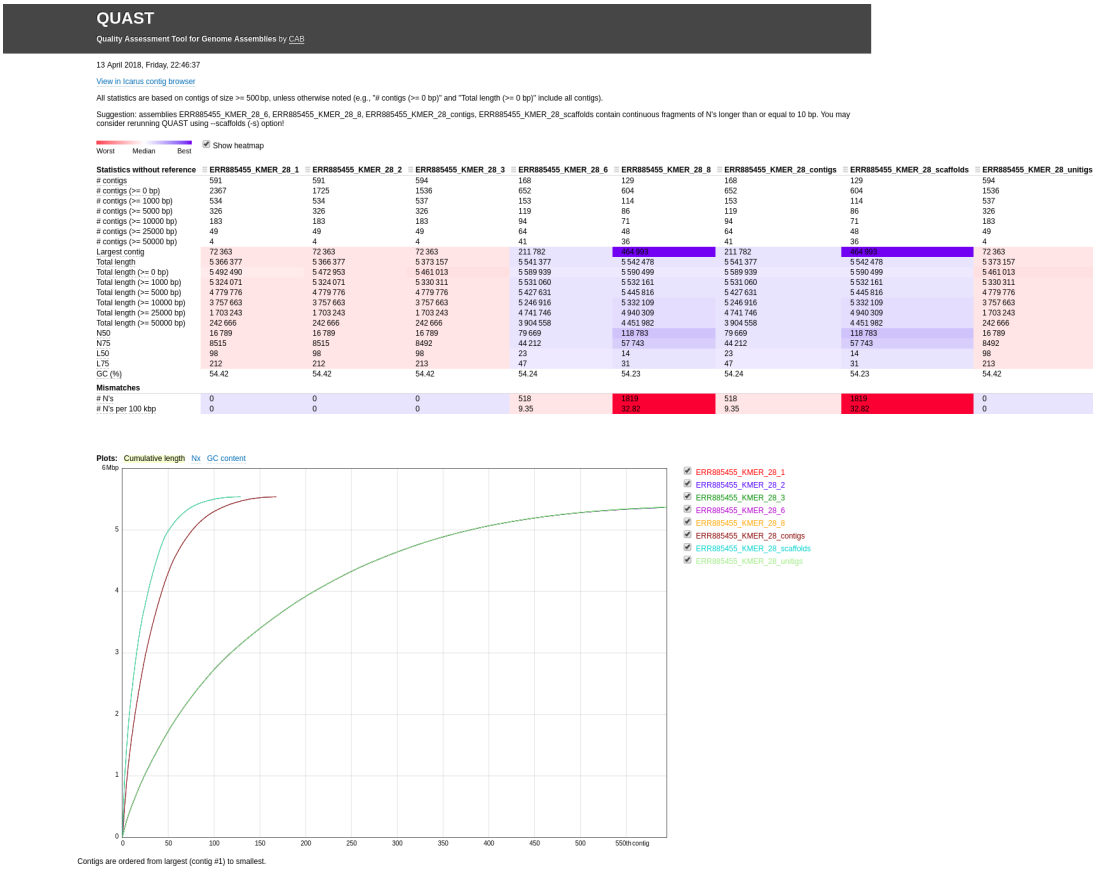


Figura I.1: Quark stats for the assembly of *Enterobacter kobei* using a K-mer = 28.

Anexo II

Workflow 2: RNA-seq de células endoteliais humanas

II.1 Workflow para mapear reads a uma referência

Illumina HiSeq2500 RNA-Seq sequences available in NCBI under accession number SRR5181508.

```
# downloading and converting raw files (reads)
fastq-dump SRR5181508

# downloading the GFF file of human genome
$ wget ftp://ftp.ensembl.org/pub/release-88/gtf/homo_sapiens/
   Homo_sapiens.GRCh38.88.gtf.gz
$ gzip -d Homo_sapiens.GRCh38.88.gtf.gz

# downloading the human chromosome 22
$ wget ftp://ftp.ensembl.org/pub/release-88/fasta/homo_sapiens/dna/
   Homo_sapiens.GRCh38.dna.chromosome.22.fa.gz
$ gzip -d Homo_sapiens.GRCh38.dna.chromosome.22.fa.gz

# Filtering with sickle
# se            sigle sequences
# --qual-type   type of quality
# --output-file output file
# -q    flag designates the minimum quality
# -l    the minimum read length
$ sickle se --fastq-file SRR5181508.fastq --qual-type sanger --output
   -file SRR5181508_FILTERED.fastq -q 30 -l 25
```

```
# Building the hisat2 index;
# -p number of processors
$ hisat2-build -p 4 Homo_sapiens.GRCh38.dna.chromosome.22.fa
   chromosome.22.hisat2.idx

# Mapping the filtered reads to the refrence;
# -p number of processors
# -q input in fastq format
# -S output file
$ hisat2 -p 2 -x chromosome.22.hisat2.idx -q SRR5181508_FILTERED.
   fastq -S SRR5181508.sam

# Analisys
# Converting formats
$ samtools view -bS SRR5181508.sam > SRR5181508.bam

# Sorting data
$ samtools sort -n SRR5181508.bam SRR5181508_SORTED.sn
$samtools view -h -o SRR5181508_SORTED.sn.sam  SRR5181508.bam

# Counting reads in features
# -m reads overlapping more than one feature
# -s strand-specific assay (default: yes)
# -o output file
# -a skip reads with alignment quality lower than x (x = default: 10)
$ htseq-count -m intersection-nonempty -s no -a 10 SRR5181508_SORTED.
   sn.bam Homo_sapiens.GRCh38.88.gtf -o SRR5181508.count
```

A partial view of the mapping can be seen in Figure II.1.

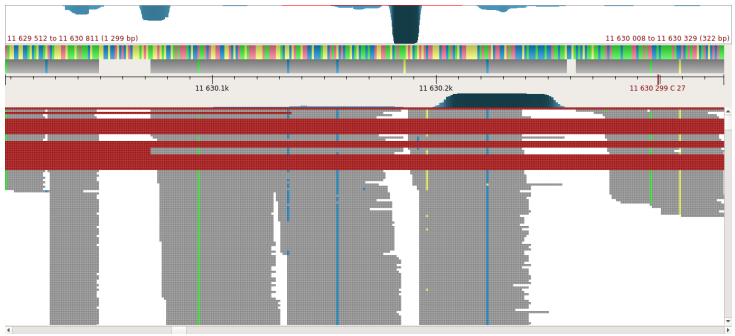


Figura II.1: A partial view of the mapping.