SciPhyloMiner: um *Workflow* para Mineração de Dados Filogenômicos de Protozários

Thaylon Guedes[1], Kary Ocaña[2], Daniel de Oliveira[1]

[1]Instituto de Computação - Universidade Federal Fluminense

[2]Laboratório de Bioinformática, Laboratório Nacional de Computação Científica

{thaylongs, danielcmo}@ic.uff.br, karyann@lncc.br

Resumo. *Uma tarefa importante na bioinformática é explorar informações evolutivas contidas em árvores filogenéticas, para a identificação padrões ou assinaturas que demonstrem a presença de determinado processo evolutivo. A presente pesquisa visa explorar as árvores filogenéticas de genes ortólogos nos genomas de protozoários no nível genômico, na procura de padrões nas sub-árvores relacionadas a processos evolutivos ou filogenômicos. No entanto, realizar a exploração e comparação de árvores de forma manual é inviável. Neste artigo foi desenvolvido um workflow científico de mineração de dados filogenômicos, o SciPhyloMiner, que apoia as comparações múltiplas de centenas ou milhares de árvores. A metodologia foi dividida em três etapas, com a etapa-1 geração de árvores, etapa-2 mineração de árvores e etapa-3 análise de árvores via consulta em bancos de dados de proveniência. Experimentos mostram que SciPhyloMiner permite a mineração de uma grande quantidade de árvores em ambientes de nuvem. Resultados de desempenho apresentam melhoras de até 94,54% no tempo de execução quando comparado com a execução sequencial, que cai de 4 dias para aproximadamente 4,85 horas.*

1. Introdução

O volume de dados genômicos disponíveis cresce em um ritmo acelerado devido às recentes melhorias nas ciências biológicas e na computação, *e.g*, as tecnologias de sequenciamento de próxima geração (SPG) e computação de alto desempenho (CAD). Atualmente, um desafio para os bioinformatas na área da filogenia e filogenômica molecular é analisar o grande volume de dados biológicos gerados, uma vez que o conhecimento sobre processos evolutivos ou relações filogenéticas na atual era pós-genômica é inferido no nível de genomas (Chen *et al*., 2012; Darling *et al*., 2014; Ocaña e Dávila, 2011; Eisen, 2003).

A filogenômica utiliza princípios da filogenia molecular na análise evolutiva de dados genômicos. Experimentos filogenômicos produzem árvores e estatísticas utilizadas para inferir a história evolutiva das espécies (Eisen, 2003). Uma tarefa importante do domínio da filogenômica é explorar informações evolutivas contidas nessas árvores, mediante a exploração e identificação de características *i.e.*, padrões ou assinaturas que demonstrem a presença de determinado processo evolutivo (Eisen, 2003). Por exemplo, determinar quais são as sub-árvores frequentes (FST) em árvores filogenômicas de famílias de proteínas, *e.g.*, ortólogos universais (OU) (Ciccarelli *et al*., 2006) pode confirmar que espécies muito relacionadas apresentam "assinaturas" filogenéticas similares em várias dessas árvores, indicando padrões ou forças seleção de

conservação semelhantes na evolução desses esses genes (Eisen, 2003). A identificação de tais padrões ajuda aos cientistas a estabelecer relações filogenéticas entre organismos, além de aceitar ou refutar hipóteses sobre a vida evolutiva desses genes ou genomas.

A procura de FST não é uma tarefa simples de ser analisada; ela pode envolver a comparação de centenas ou milhares de árvores (número que pode aumentar de acordo com a quantidade de genomas avaliados). Comumente, cientistas inferem esse conhecimento de forma manual. Entretanto, este processo é laborioso e propenso a erros, devido ao grande volume de dados. Acreditamos que técnicas de Mineração de Dados (MD) (Wang, 2005) são capazes de auxiliar os cientistas na extração de padrões e informações úteis desse grande volume de dados. Uma vez que tais técnicas são capazes de extrair conhecimento não trivial, implícito, previamente desconhecido e úteis dos dados (Wang, 2005). A identificação de FST em árvores filogenômicas utilizando algoritmos de MD pode potencialmente descobrir mecanismos genéticos subjacentes a uma doença, resumir regras de agrupamento para múltiplas sequências de DNA ou proteínas, *etc.*, que são problemas emergentes na área (Clark, 2006).

Experimentos científicos em larga escala que envolvem tarefas de MD em análises filogenômicas são comumente executados centenas ou milhares de vezes e a análise dos resultados deve considerar (e cruzar dados de) todas essas avaliações. Esses experimentos podem ser apoiados por técnicas como os *workflows* científicos (Taylor *et al*., 2007) e sistemas de gerência de *workflows* científicos (SGWfC), responsáveis por executar tais *workflows.* Como muitos desses *workflows* são de larga escala, eles precisam ser executados em ambientes de CAD, o que insere mais uma complexidade. Outros *workflows* na área de bioinformática já foram propostos anteriormente (Ocaña *et al*., 2014; Ocaña *et al*., 2014; Ocaña *et al*., 2013; Oliveira *et al*., 2013), porém nenhum que focasse na mineração de árvores filogenômicas.

Este artigo propõe o SciPhyloMiner, um *workflow* científico para mineração de dados filogenômicos, que visa a identificação de padrões comuns *i.e.* FST nas topologias de árvores a nível genômico. O SciPhyloMiner estende um *workflow* já existente e consolidado (SciPhy) (Ocaña *et al*., 2011). No SciPhyloMiner, as atividades do SciPhy permanecem, mas atividades específicas de mineração de dados são adicionadas. O SciPhyloMiner foi modelado e executado com o SGWfC SciCumulus (Oliveira *et al*., 2010). O SciPhyloMiner permite que todas as árvores produzidas (formato Newick), assim como outros dados de proveniência sejam minerados e armazenados no banco de dados de proveniência do SciCumulus. Desta forma, o SciPhyloMiner permite a análise desses dados via consultas SQL ao banco. Assim, os cientistas são capazes, por exemplo, de rastrear a linhagem taxonômica de um táxon dentre várias árvores apenas consultando o banco de dados *e.g.*, determinando os nós filhos de um pai dentro de árvores ou pesquisando sub-árvores compartilhadas entre diferentes árvores. Na avaliação da abordagem proposta, analisamos um conjunto de 800 árvores filogenéticas de genes e de protozoários; as quais foram construídas com 4 programas de filogenia diferentes; dados contidos nessas árvores foram minerados e a presença de FST foi realizada em uma análise a nível filogenômico.

Esse artigo está organizado em 5 seções, além desta introdução. A Seção 2 discute os trabalhos relacionados. A Seção 3 apresenta o SciPhyloMiner. A Seção 4 apresenta a análise dos resultados experimentais. Finalmente, a Seção 5 conclui o artigo.

2. Trabalhos Relacionados

No melhor de nosso conhecimento, não existem *workflows* que propõem aplicação de técnicas de MD em árvores filogenômicas de forma estruturada. Dessa forma, apresentamos aplicações existentes que mineram árvores filogenômicas e que facilmente poderiam ser incorporadas ao SciPhyloMiner. Um dos algoritmos mais usados na identificação de padrões comuns, o FST, foi implementado pelo algoritmo Evominer (Deepak *et al*., 2014), acoplado ao *workflow* SciPhyloMiner. Deepak *et al.* (2014) apresentam uma revisão sobre os métodos mais usados na identificação da informação filogenética. Dentre deles, o TreeMiner, o FREQT, o Chopper e XSpanner, o Unot, o uFreqt, o CMTreeMiner, o DRYADE, o Phylominer e o HybridTreeMiner. Nessa revisão, os autores afirmam que o único algoritmo de código acessível foi o Phylominer, que e foi usado para as comparações de desempenho, onde o Evominer se mostrou 100 vezes mais rápido.

O PhyloFinder (Chen *et al*., 2008) é uma aplicação baseada na *Web* para a procura de árvores em banco de dados (TreeBASE); a entrada é uma árvore que é comparada contra o TreeBase via consulta por palavra-chave ou nome do táxon, podendo navegar pelos ancestrais e filhos desse táxon. O PhyloSift (Darling *et al*., 2014) é uma ferramenta para análises filogenéticas em genomas focado na anotação e análise funcional e taxonômica de metagenomas. O EnsemblCompara GeneTrees (Vilella *et al*., 2009) é um *pipeline* computacional para agrupamento, Alinhamento Múltiplo de Sequência (AMS) e geração de árvore, incluindo o tratamento de grandes famílias de genes.

3. Abordagem Proposta: o SciPhyloMiner

O SciPhyloMiner é um *workflow* para análise de árvores filogenômicas baseado em técnicas de MD. Ele é composto por 9 atividades principais, distribuídas em três *sub-workflows*, como apresentado na Figura 1. O primeiro *sub-workflow* de filogenia, o SciPhy (Ocaña *et al*., 2011), tem 4 atividades: (1) construção de AMS, (2) conversão de AMS para o formato PHYLIP, (3) eleição do melhor modelo evolutivo, e (4) construção das árvores filogenéticas. Os programas que executam respectivamente essas atividades são MAFFT, ReadSeq, ModelGenerator e RAxML. Na atividade de construção de árvores, o SciPhyloMiner utiliza outros 3 programas: PhyML, Garli e MrBayes. As árvores filogenéticas produzidas são usadas como entrada para a mineração de dados filogenômicos nos *sub-workflows* que se seguem, que usam como principal programa o Dendropy. O *sub-workflow* 2 é composto pelas atividades (5) junção de árvores para redução, (6) mineração de árvores e (7) representação de FST. O *sub-workflow* 3 realiza tanto a mineração quanto a análise dos dados com as atividades (8) junção de árvores para análise sintática e (9) análise sintática de árvores para representação. A seguir detalhamos cada uma das atividades do SciPhyloMiner.

A atividade 1 recebe o arquivo multi-fasta e constrói o AMS com o MAFFT. A atividade 2 executa o ReadSeq para converter o AMS ao formato PHYLIP. A atividade 3 executa o ModelGenerator para a escolha do melhor modelo evolutivo. A atividade 4 é responsável pela construção das árvores usando algoritmos de máxima verossimilhança (MV) com o RAxML, PhyML e Garli e inferência Bayesiana (IB) com o MrBayes (outros programas como Phylip, Weighbor, Beast, *etc*., podem também ser acoplados). Esta atividade usa como entrada tanto o modelo evolutivo do ModelGenerator quanto o AMS do MAFFT.

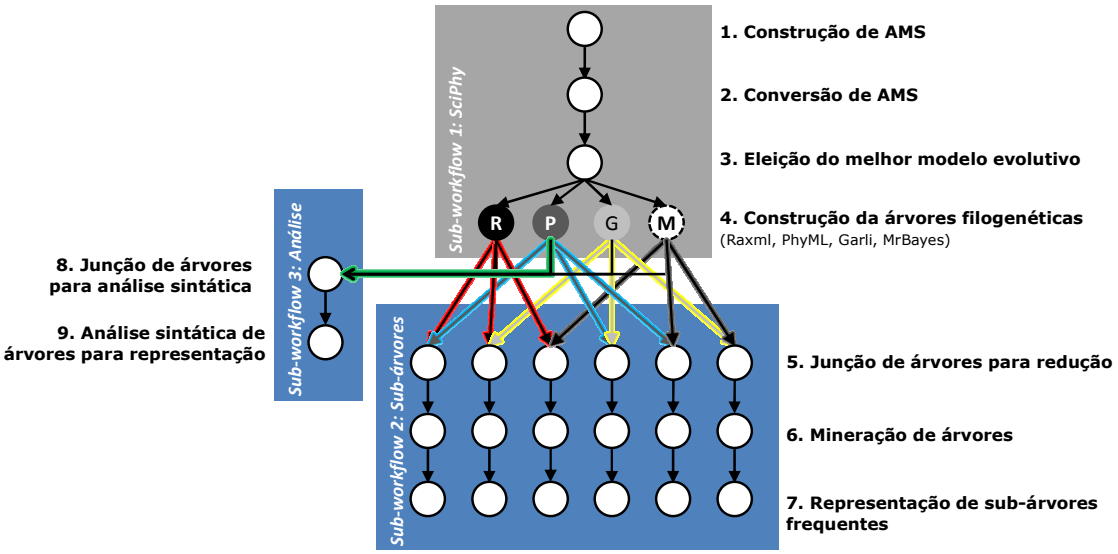

Figura 1. Especificação Conceitual do SciPhyloMiner

No SciPhyloMiner, cada uma das atividades de filogenia é executada em paralelo para cada entrada (arquivo multi-fasta). Mas o mapeamento, comparação de todos os programas e a junção/filtragem de resultados não é uma tarefa trivial de ser realizada, pois é necessário realizar uma combinação 2 a 2 entre todos os métodos de geração de árvores filogenéticas (*shuffle)*. Assim, consultas SQL ao banco de proveniência do SciCumulus (para atividades dos *sub-workflows* 2 e 3) apoiam essas tarefas. Por exemplo, para cada arquivo de entrada multi-fasta processados com 4 programas filogenéticos (R: RAxML, P: PhyML, G: Garli e M: MrBayes), como descrito na Figura 1, será gerado um total de 6 comparações. Então, para o nosso conjunto de dados de 200 arquivos multi-fasta de entrada, 800 árvores filogenômicas finais de consenso são comparadas (uma para cada programa filogenético). No total, 1.200 comparações são realizadas entre essas árvores, gerando 1.200 arquivos intermediários com estatísticas. Neste trabalho consideramos somente as árvores obtidas ao final da execução do experimento. Entretanto, muitos outros arquivos, como árvores intermediárias e arquivos com estatísticas, também são gerados, e que embora não sejam minerados, podem conter informações úteis para o bioinformata que ajudem a reforçar uma análise.

Na atividade 5, todos os arquivos com árvores filogenéticas são mapeados e organizados aos pares em uma tabela que facilita futuras consultas de dados de domínio, usada pela atividade 6. A atividade 6 executa o programa Dendropy v4.0 usando Python 2.7 e os algoritmos de MD *fingerprint* (Karp e Rabin, 1987) e Evominer (Deepak *et al.*, 2014). O *fingerprint* trata as folhas das árvores como uma *string*. Os *fingerprints* de cada sub-árvore são classificados “frequentes” em uma tabela *hash*, o que facilita a contagem e verifica a frequência mínima das sub-árvores candidatas em uma árvore de *k* folhas. O Evominer lê cada tabela contendo os pares de árvores, calcula a FST e gera dois arquivos, o *totalFST* com a quantidade de folhas sobrepostas em todas as árvores e o *maxFST* com o valor máximo de folhas sobrepostas em todas as árvores. A frequência mínima requerida para o cálculo de FST foi fixado neste artigo com o padrão de 0,5.

A atividade 7 é responsável por transformar o formato Newick das árvores filogenéticas em um formato tabular. O formato tabular é, então, armazenado e

consultado no banco de dados, tornando a tabela compreensível também para os cientistas, conforme ilustrado na figura 3. As atividades finais 8 e 9 podem ser usadas em outros processos como na análise estatística ou visualização que exigem tratar e explorar árvores; a atividade 8 analisa árvores e a atividade 9 representa as árvores por meio de tabelas.

4. Avaliação Experimental

Esta seção, apresenta uma avaliação dos resultados obtidos pela nas execuções do SciPhyloMiner, tanto sob a perspectiva biológica quanto no aspecto computacional. Do ponto de vista biológico, destacamos o uso e vantagens na exploração de topologias de árvores em um conjunto de genes denominados ortólogos universais (OU) presentes em genomas protozoários (Ocaña e Dávila, 2011). O objetivo é detectar as FST apresentadas nas árvores que podem sugerir uma assinatura filogenética representativa de tais genes. Esta informação é útil para a categorização de novas espécies de protozoários que possam ser sequenciados por metodologias genômicas/metagenômicas. Do ponto de vista computacional, analisamos o desempenho da execução paralela do SciPhyloMiner usando SciCumulus em nuvens de computadores. Implementamos todas as aplicações de bioinformática MAFFT, ReadSeq, RAxML, PhyML, Garli, MrBayes, Dendropy, bibliotecas e componentes do SciCumulus no ambiente do Amazon EC2.

A Amazon EC2 fornece diferentes tipos de máquinas virtuais (VM). A VM c3.8xlarge (60 GB de RAM, 32 núcleos) foi a escolhida para as execuções do *workflow*. Esta VM usa o Linux Cent OS 5 (64 *bits*) como sistema operacional e foi configurada com os programas de bioinformática e bibliotecas necessários. Além de permitir o acesso via SSH. A imagem utilizada na VM em questão foi a ami-2579d333, a qual está disponibilizada na nuvem.

Para executar SciPhyloMiner em paralelo, nossos experimentos usam como entrada um conjunto de dados de arquivos multi-fasta de sequências de proteínas extraídas do NCBI (Ocaña e Dávila, 2011). Este conjunto de dados é formado por 200 arquivos multi-fasta, cada arquivo relacionado a um determinado gene OU encontrado nas diferentes espécies de genomas de protozoários. Para realizar a análise filogenômica, cada arquivo multi-fasta é processado usando as seguintes versões dos programas: MAFFT versão 6.857, ReadSeq versão 2.1.26, ModelGenerator versão 0.85, RAxML-7.2.8-ALPHA, PhyML 3.1, Garli 2.0.1.1067, MrBayes v3.2.6 x64 e Dendropy v4. 0 com Python 2.7.

A Figura 2 apresenta as árvores construídas com os programas de filogenia PhyML, RAxML, Garli e MrBayes. T1-T4 mostram as topologias das árvores filogenéticas visualizadas no programa Mega. A Figura 3 apresenta o formato de tabela legível que pode ser armazenada e consultada via SQL no banco de proveniência do SciCumulus. O gene analisado apresenta árvores muito semelhantes (de T1 a T4) baseado na topologia, comprimento de ramos nas árvores, o *bootstrap* e a probabilidade *a posteriori*. O comprimento do ramo se refere à quantidade de mudanças genéticas ou tempo decorrido. O *bootstrap* e a probabilidade *a posteriori* são técnicas estatísticas de confiabilidade das estimativas das relações evolutivas corretas. Neste trabalho, eles apresentaram valores altos (>95) indicando relações consistentes entre os táxons.

Em relação à avaliação de desempenho do *workflow*, executamos o SciPhyloMiner com o SciCumulus na nuvem da Amazon EC2. Primeiramente, medimos o desempenho do SciPhyloMiner quando executado em uma VM com um

núcleo para medir a otimização local antes de aumentar o número de núcleos. A seguir, medimos o desempenho do SciPhyloMiner com uma VMs c3.8xlarge, que possui 32 núcleos virtuais.

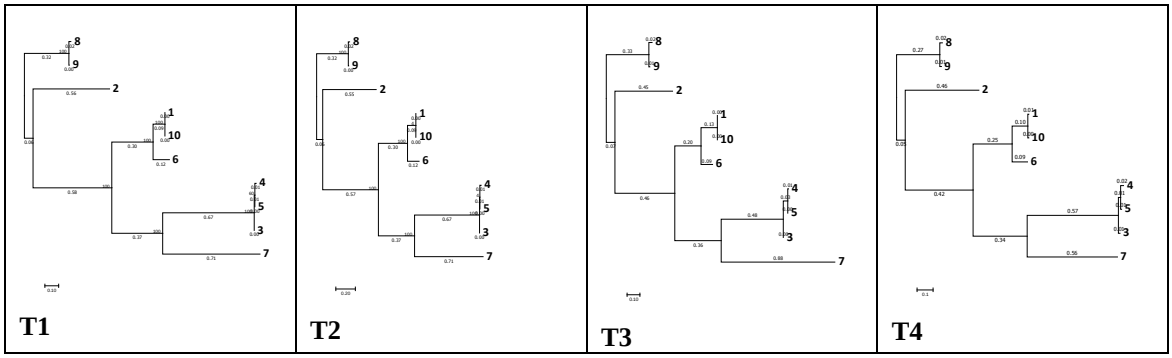

Figura 2. Representação de árvores

```
[root@ip-10-141-161-106 0]# more ORTHOMCL337_ORTHOMCL337_1.treetable
0.5     ORTHOMCL337_ORTHOMCL337_1_ORTHOMCL337_ORTHOMCL337_1.maxFST.evominer.tree_0      3       15      11
0.5     ORTHOMCL337_ORTHOMCL337_1_ORTHOMCL337_ORTHOMCL337_1.maxFST.evominer.tree_0      None    2       1
0.5     ORTHOMCL337_ORTHOMCL337_1_ORTHOMCL337_ORTHOMCL337_1.maxFST.evominer.tree_0      8       3       1
0.5     ORTHOMCL337_ORTHOMCL337_1_ORTHOMCL337_ORTHOMCL337_1.maxFST.evominer.tree_0      1       13      9
0.5     ORTHOMCL337_ORTHOMCL337_1_ORTHOMCL337_ORTHOMCL337_1.maxFST.evominer.tree_0      5       16      16
0.5     ORTHOMCL337_ORTHOMCL337_1_ORTHOMCL337_ORTHOMCL337_1.maxFST.evominer.tree_0      9       4       1
0.5     ORTHOMCL337_ORTHOMCL337_1_ORTHOMCL337_ORTHOMCL337_1.maxFST.evominer.tree_0      10      14      9
0.5     ORTHOMCL337_ORTHOMCL337_1_ORTHOMCL337_ORTHOMCL337_1.maxFST.evominer.tree_0      None    16      11
0.5     ORTHOMCL337_ORTHOMCL337_1_ORTHOMCL337_ORTHOMCL337_1.maxFST.evominer.tree_0      None    1       None
0.5     ORTHOMCL337_ORTHOMCL337_1_ORTHOMCL337_ORTHOMCL337_1.maxFST.evominer.tree_0      6       10      7
0.5     ORTHOMCL337_ORTHOMCL337_1_ORTHOMCL337_ORTHOMCL337_1.maxFST.evominer.tree_0      4       17      16
0.5     ORTHOMCL337_ORTHOMCL337_1_ORTHOMCL337_ORTHOMCL337_1.maxFST.evominer.tree_0      None    9       7
0.5     ORTHOMCL337_ORTHOMCL337_1_ORTHOMCL337_ORTHOMCL337_1.maxFST.evominer.tree_0      7       12      8
0.5     ORTHOMCL337_ORTHOMCL337_1_ORTHOMCL337_ORTHOMCL337_1.maxFST.evominer.tree_0      None    11      8
0.5     ORTHOMCL337_ORTHOMCL337_1_ORTHOMCL337_ORTHOMCL337_1.maxFST.evominer.tree_0      None    8       5
0.5     ORTHOMCL337_ORTHOMCL337_1_ORTHOMCL337_ORTHOMCL337_1.maxFST.evominer.tree_0      None    5       2
0.5     ORTHOMCL337_ORTHOMCL337_1_ORTHOMCL337_ORTHOMCL337_1.maxFST.evominer.tree_0      2       6       2
0.5     ORTHOMCL337_ORTHOMCL337_1_ORTHOMCL337_ORTHOMCL337_1.maxFST.evominer.tree_0      None    7       5
```

Figura 3 Representação de uma árvore em forma de tabela

Na Figura 4 são apresentadas as medições do tempo de execução total (TET) em horas de SciPhyloMiner. Esse tempo diminui, em todos os casos, quando o SciCumulus permite que mais núcleos sejam utilizados na execução. Por exemplo, o TET foi reduzido de 88,90 horas (usando um único núcleo virtual) para 4,85 horas (usando 32 núcleos virtuais), o que significa melhorias de desempenho de até 94,54%. Para avaliar o comportamento do ganho de desempenho de acordo com o número de *cores*, utilizamos a métrica de *speedup*. A execução do SciPhyloMiner usando 16 núcleos virtuais levou a um *speedup* de 14,39.

Além da análise de desempenho e biológica, realizamos consultas analíticas na base que permitem ao cientista consultar de fato o resultado da mineração. A Tabela 1 apresenta uma consulta que calcula o número máximo de FST encontrados pela comparação de todos os programas de filogenia (A) e o resultado dessa consulta (B), usando uma entrada de gene OU. O interessante deste resultado é a diferença entre os programas PhyML *versus* Garli (ambos baseados no algoritmo MV), o mesmo comportamento foi obtido para as demais entradas (OU). Enquanto todas as outras comparações convergiram em um valor FST 2, o *phyml_garli* apresentou FST 1. Este resultado leva ao especialista levantar possíveis causas que levaram a estes diferentes valores *e.g.*, características biológicas das entradas ou a parametrização/programa. Podemos afirmar baseados nos dados obtidos neste experimento, que esses dois programas foram os que mais divergiram. Uma causa pode estar relacionada à otimização e refinamento que o próprio algoritmo dentro do Garli faz na parte da procura de topologias, o que é calculado e configurado de uma maneira mais simples no PhyML. Mas por outro lado, se o objetivo é uma análise exploratória, a eleição do PhyML pode se tornar uma boa alternativa, devido à rapidez na execução, especialmente na reconstrução de árvores muito grandes.

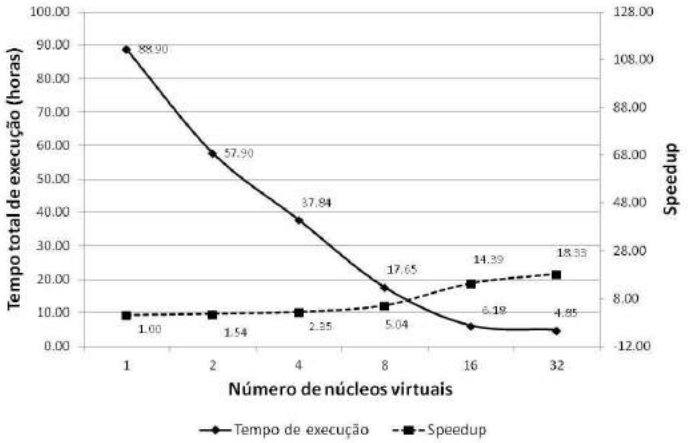

Figura 4. Tempo total de execução e *speedup* do *workflow* SciPhyloMiner

Tabela 1 Consulta (A) e resultado da consulta (B) de sub-árvores frequentes (FST)

```
SELECT ewf.tagexec, evocount.maximum_fsts
FROM (SELECT ev.ewkfid, count(*) as maximum_fsts
     FROM (SELECT distinct evo.ewkfid,
evo.tree_reference FROM
sciphytreeminer.oevotreeparser_{metodo1_metodo2}
as evo) as ev

     GROUP BY ev.ewkfid) as evocount inner join
eworkflow as ewf on ewf.ewkfid = evocount.ewkfid
```

Programas	Tagexec	Max.FST
mrbayes_garli	1wftreeminer	2
phyml_garli	1wftreeminer	1
phyml_mrbayes	1wftreeminer	2
raxml_garli	1wftreeminer	2
raxml_mrbayes	1wftreeminer	2
raxml_phyml	1wftreeminer	2

5. Conclusões

Neste artigo, apresentamos um experimento de mineração de dados filogenômicos modelados no *workflow* científico SciPhyloMiner que foi executado em um ambiente de nuvem pública (Amazon EC2) usando o SciCumulus para fornecer as capacidades de paralelismo necessárias. Por meio do acoplamento de *workflows* científicos, SGWfC e abordagens especializadas para paralelizar os *workflows*, o SciPhyloMiner permite: (i) gerenciar os experimentos de mineração para filogenômica, (ii) extrair e consultar a proveniência de dados do experimento, com foco em topologias de árvores e (iii) se beneficiar do poder de processamento das nuvens.

O desempenho e a escalabilidade da execução do SciPhyloMiner envolveram milhares de tarefas e arquivos de dados do experimento. Analisando o desempenho geral, podemos afirmar que o desempenho do *workflow* melhorou em até 94,54% quando comparado a uma execução sequencial do mesmo. A análise de *speedup* mostrou que quando aumentamos o número de MVs, o *speedup* também aumentou com uma leve degradação (aceitável). No entanto, o custo/benefício de adicionar MVs precisa considerar vários aspectos, como o número e características das atividades do *workflow* a serem executadas, o custo monetário e os ganhos esperados.

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

Referências Bibliográficas

Chen, D., Burleigh, J.G., Bansal, M.S., Fernández-Baca, D., 2008. PhyloFinder: An intelligent search engine for phylogenetic tree databases. BMC Evolutionary Biology 8, 90.

Chen, M., Zou, M., Yang, L., He, S., 2012. Basal Jawed Vertebrate Phylogenomics Using Transcriptomic Data from Solexa Sequencing. PLoS ONE 7, e36256.

Ciccarelli, F.D., Doerks, T., von Mering, C., Creevey, C.J., Snel, B., Bork, P., 2006. Toward automatic reconstruction of a highly resolved tree of life. Science 311, 1283–1287.

Clark, A.G., 2006. Genomics of the evolutionary process. Trends in Ecology & Evolution 21, 316–321.

Darling, A.E., Jospin, G., Lowe, E., Matsen, F.A., Bik, H.M., Eisen, J.A., 2014. PhyloSift: phylogenetic analysis of genomes and metagenomes. PeerJ 2, e243.

Dávila, Kary A. C. S. Ocaña, 2011. Phylogenomics-Based Reconstruction of Protozoan Species Tree. EBO 107.

Deepak, A., Fernández-Baca, D., Tirthapura, S., Sanderson, M.J., McMahon, M.M., 2014. EvoMiner: frequent subtree mining in phylogenetic databases. Knowledge and Information Systems 41, 559–590.

Eisen, J.A., 2003. Phylogenomics: Intersection of Evolution and Genomics. Science 300, 1706–1707.

Karp, R.M., Rabin, M.O., 1987. Efficient randomized pattern-matching algorithms. IBM Journal of Research and Development 31, 249–260.

Ocaña, K., Benza, S., Oliveira, D., Dias, J., Mattoso, M., 2014. Exploring Large Scale Receptor-Ligand Pairs in Molecular Docking Workflows in HPC Clouds, in: IEEE 28th International Parallel & Distributed Processing Symposium Workshops (HiComb 2014). IPDPS, Phoenix, Arizona, USA, pp. 536–545.

Ocaña, K.A.C.S., de Oliveira, D., Dias, J., Ogasawara, E., Mattoso, M., 2013. Designing a parallel cloud based comparative genomics workflow to improve phylogenetic analyses. Future Generation Computer Systems 29, 2205–2219.

Ocaña, K.A.C.S., Oliveira, D., Silva, V., Benza, S., Mattoso, M.L.Q., 2014. Exploiting the Parallel Execution of Homology Workflow Variants in HPC Compute Clouds, in: 4th International Workshop on Cloud Computing and Scientific Applications (CCSA 2014), France.

Ocaña, K., Oliveira, D. de, Ogasawara, E., Dávila, A., Lima, A., Mattoso, M., 2011. SciPhy: A Cloud-Based Workflow for Phylogenetic Analysis of Drug Targets in Protozoan Genomes, Adv. Bioinformatics Computational Biology, LNCS. Springer, pp. 66–70.

Oliveira, D., Ocaña, K.A.C.S., Ogasawara, E., Dias, J., Gonçalves, J., Baião, F., Mattoso, M., 2013. Performance evaluation of parallel strategies in public clouds: A study with phylogenomic workflows. Future Generation Computer Systems 29, 1816–1825.

Oliveira, D., Ogasawara, E., Baião, F., Mattoso, M., 2010. SciCumulus: A Lightweight Cloud Middleware to Explore Many Task Computing Paradigm in Scientific Workflows, International Conference on Cloud Computing. Washington, DC, USA, pp. 378–385.

Taylor, I.J., Deelman, E., Gannon, D.B., Shields, M., 2007. Workflows for e-Science: Scientific Workflows for Grids, 1st ed. Springer.

Vilella, A.J., Severin, J., Ureta-Vidal, A., Heng, L., Durbin, R., Birney, E., 2009. EnsemblCompara GeneTrees: Complete, duplication-aware phylogenetic trees in vertebrates. Genome Res. 19, 327–335.

Wang, J.T.L., 2005. Data mining in bioinformatics. Springer, London.