COPPE/UFRJ

UM SISTEMA DE RECOMENDAÇÃO PARA COMPOSIÇÃO DE WORKFLOWS

Frederico Tosta de Oliveira

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

Orientadores: Marta Lima de Queirós Mattoso

Leonardo Gresta Paulino Murta

Rio de Janeiro

Junho de 2010

UM SISTEMA DE RECOMENDAÇÃO PARA COMPOSIÇÃO DE WORKFLOWS

Frederico Tosta de Oliveira

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

Examinada por:

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

_____
Prof. Leonardo Gresta Paulino Murta, D. Sc.

_____
Prof. Cláudia Maria Lima Werner, D. Sc.

_____
Prof. Alexandre Gonçalves Evsukoff, D. Sc.

RIO DE JANEIRO, RJ - BRASIL
JUNHO DE 2010

Oliveira, Frederico Tosta de

Sistema de Recomendação para Composição de Workflows/ Frederico Tosta de Oliveira. – Rio de Janeiro: UFRJ/COPPE, 2010.

XI, 80p.: il.; 29,7 cm.

Orientadores: Marta Lima de Queirós Mattoso
Leonardo Gresta Paulino Murta.

Dissertação (mestrado) – UFRJ/ COPPE/ Programa de Engenharia de Sistemas e Computação, 2010.

Referencias Bibliográficas: p. 72 - 77.

1. Sistemas de Recomendação. 2. Workflows.

I. Mattoso, Marta Lima de Queirós, *et. al.*

II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia de Sistemas e Computação. III. Titulo.

À minha esposa, Bárbara, e ao meu filho, Guilherme.

Agradecimentos

À Deus, por iluminar-me e abençoar-me sempre.

A minha querida e amada esposa, Bárbara dos Santos Rosa, pelo seu amor, carinho, torcida, companheirismo durante esta fase de nossas vidas, paciência e compreensão nas ocasiões em que não pude dar-lhe a merecida atenção.

Aos meus pais, Sandoval Borges de Oliveira Filho e Alzira Debora Tosta de Oliveira, pelo amor, carinho e por me ensinarem a importância do estudo e do conhecimento.

À minha irmã, Barbara Tosta de Oliveira, amiga que sempre torce por mim, acompanha e incentiva meus estudos.

Aos meus orientadores, Marta Lima de Queirós Mattoso e Leonardo Gresta Paulino Murta, nos quais admiro muito e sou profundamente agradecido por terem me dado a oportunidade de aprendizado concedida durante o mestrado, terem se dispostos a me orientar e me auxiliar nesta jornada através de conselhos sensatos e oportunos e tempo dedicado ao acompanhamento deste trabalho.

Os amigos, Sérgio Serra e Raimundo Macário, pelos incansáveis incentivos e dicas.

Ao Exército Brasileiro pelo auxílio que me foi concedido.

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

UM SISTEMA DE RECOMENDAÇÃO PARA COMPOSIÇÃO DE WORKFLOWS

Frederico Tosta de Oliveira

Junho/2010

Orientadores: Marta Lima de Queirós Mattoso
Leonardo Gresta Paulino Murta

Programa: Engenharia de Sistemas e Computação

Com a popularização dos sistemas de gerenciamento de workflows, houve a necessidade de se disponibilizar mais funcionalidades, tornado estes sistemas cada vez mais complexos. Encontrar as dependências entre programas e serviços disponíveis, sem o suporte de uma ferramenta, não é uma tarefa trivial e em muitos casos, se transforma em uma barreira para a construção de análises e modelos mais sofisticados.

Uma das frentes de pesquisa aplicada à workflows é a reutilização de workflows. Esta reutilização visa tirar vantagem da existência de workflows previamente elaborados para se compor novos workflows, sugerindo as combinações de programas mais frequentes.

No domínio de comércio eletrônico, os sistemas de recomendação aplicam técnicas de mineração de dados consagradas para ajudar os usuários a encontrar os itens que eles gostariam de adquirir, predizendo uma lista dos itens que usuário provavelmente mais gosta. Esta dissertação traça um paralelo entre o domínio de comércio eletrônico e o domínio de workflows, de forma a permitir a utilização de uma técnica de mineração de dados chamada regras de associação, utilizada no comércio eletrônico, na composição de novos workflows.

Para esta dissertação, foi desenvolvida uma abordagem e um protótipo para demonstrar a eficácia e eficiência de sistemas de recomendação para workflows baseados em regras de associação.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

A RECOMMENDATION SYSTEM FOR WORKFLOW COMPOSITION

Frederico Tosta de Oliveira

Junho/2010

Advisors: Marta Lima de Queirós Mattoso
Leonardo Gresta Paulino Murta

Department: Computer and Systems Engineering

With the popularity of workflow management systems, there was a need to provide more functionality and these systems become more complex. To find the dependencies between programs and services available, without the support of a tool, is not a trivial task and in many cases becomes a barrier to building more sophisticated models and analysis.

One of the areas of research applied to workflows is the reuse of workflows. This reuse aims to take advantage of the existence of pre-designed workflows to compose new workflows, suggesting the combinations of programs that occur more often.

In the domain of e-commerce, recommender systems apply data mining techniques to help users find items they would like to purchase, predicting a list of items frequently liked. This dissertation draws a parallel between the domain of e-commerce and the dominance of workflows, to allow the use of a data mining technique called association rules, used in e-commerce, in the composition of new workflows.

For this dissertation, we developed an approach and a prototype to demonstrate the effectiveness and efficiency of recommendation systems for workflows based on association rules.

Índice

Capítulo 1 - Introdução 1

1.1. Motivação 1

1.2. Objetivo 2

1.3. Organização 3

Capítulo 2 - Recomendação em Workflows Científicos 5

2.1. Introdução 5

2.2. Sistemas de Recomendação 5

2.3. Workflow Científico 7

2.4. Sistemas de Recomendação em Workflows Científicos 10

2.5. Considerações Finais 19

Capítulo 3 - Descoberta do Conhecimento em Base de Dados 21

3.1. Introdução 21

3.2. Regras de Associação 24

3.3. Mineração Sequencial 28

3.4. Algoritmo para Mineração Sequencial 30

3.5. Algoritmo Selecionado 33

3.6. Considerações Finais 34

Capítulo 4 - O Sistema de Recomendação Proposto 35

4.1. Introdução 35

4.2. Visão Geral da Abordagem 35

4.3. Adaptação do Problema de Mineração de Dados 36

4.4. Fase de Preparação 39

4.4.1. Extração dos Caminhos Máximos 40

4.4.2. Separação dos Caminhos por Domínio 41

4.4.3. Atribuição de Pesos 42
4.4.4. Aplicação do Algoritmo de Mineração Sequencial 43
4.5. Fase de Consulta e Recomendação 44
4.6. Considerações finais 47
Capítulo 5 - Implementação e Avaliação 48
5.1. Introdução 48
5.2. Detalhamento Técnico da Fase de Preparação 48
5.3. Detalhamento Técnico da Fase de Consulta e Recomendação 52
5.4. Avaliação da Abordagem Proposta 53
5.4.1. Avaliação da Fase de Preparação 55
5.4.2. Avaliação da Fase de Consulta e Recomendação 58
5.4.3. Casos Particulares de Recomendação 65
5.5. Considerações Finais 67
Capítulo 6 - Conclusões 68
6.2. Limitações 70
6.3. Trabalhos futuros 70
Referências Bibliográficas 72
Apêndice A – Arquivo de Entrada para o Algoritmo PLWAP 78
Apêndice B – Arquivo Gerado pelo Algoritmo PLWAP 79
Apêndice C – Arquivo Resultante de uma Recomendação 80

Índice de Figuras

Figura 2-1 – Interface do VisTrails 10

Figura 2-2 - Usando SmartLink para conectar os módulos VTKQuadric e VTKContourFilter. *Fonte: (TELEA, 2000)* 12

Figura 2-3 - a) Porta "propriedade" do VTKActor selecionada. b) Popup menu com as opções de módulos. c) Modulo VTKProperty conectado. *Fonte: (TELEA, 2000)* 13

Figura 2-5 - As predições são refinadas iterativamente. *Fonte: (KOOP, 2008).* 15

Figura 2-7 - Abordagem proposta por Oliveira (2008) 17

Figura 2-8 - Exibição de múltiplas recomendações no SmartLink. *Fonte:* (TELEA, 2000). 18

Figura 3-1 - Visão geral dos passos que compõem o processo de KDD. *Fonte: (FAYYAD, 1996b).* 22

Figura 3-2 - (a) Banco de dados de transações ordenado por id cliente e *timestamp*. (b) Banco de dados com as sequências dos clientes. (c) Conjunto de respostas. *Fonte:* (AGRAWAL, 1995) 30

Figura 3-3 - Tempo de execução para um conjunto de dados com diferentes suporte ((a) -banco de dados pequeno, (b) – banco de dados médio e (c) – banco de dados grande). *Fonte:* (EZEIFE, 2005b)..... 33

Figura 4-1 - Visão Geral da Abordagem 36

Figura 4-2 - Sequência de compras de um cliente na forma gráfica. 37

Figura 4-3 - Workflow composto por sete tarefas (a) e seus caminhos (b). 38

Figura 4-4- Visão Geral da Fase de Preparação 40

Figura 4-5 - Caminhos separados por domínio. 41

Figura 4-6 - Caminhos separados por execução. 43

Figura 4-7 - Etapas da Consulta e Recomendação. 45

Figura 4-8 - Consultas (a) e sub-consultas (b). 46
Figura 4-9 - Recomendações (a) e workflow final (b). 46
Figura 5-1 - Diagrama entidade-relacionamento da abordagem. 49
Figura 5-2 - Workflow com identificador nas tarefas. 50
Figura 5-3 - Tabelas preenchidas com workflow exemplo. 50
Figura 5-4 - Snapshot da fase de preparação para a base utilizada. 56
Figura 5-5- Gráficos de utilização de tempo e memória da fase de preparação. 57
Figura 5-6 - Gráficos de utilização de tempo e memória para carregar as bases preparadas. 59
Figura 5-7 - Calculando a semelhança entre caminhos recomendados. 61
Figura 5-8 - Construção de um workflow com recomendação no VisComplete. 62
Figura 5-9 - Caso particular de recomendação. 66

Capítulo 1 - Introdução

1.1.Motivação

Os experimentos científicos são tipicamente compostos por programas em *pipeline* que manipulam uma enorme quantidade de dados. Normalmente, esses experimentos são construídos de forma manual, conectando entradas e saídas de programas formando um fluxo de execução (CRUZ, 2008a). As saídas são analisadas e, de acordo com os resultados, parâmetros são acertados, o *pipeline* é re-executado, os programas são substituídos e possivelmente uma execução parcial do *pipeline* é necessária. Atualmente, muitos *pipelines* de bioinformática são modelados e executados em linguagens de script, como perl e python, pelos próprios cientistas, mas como são fortemente ligados ao código fonte são difíceis de gerar, gerenciar e manter.

Na computação, a tecnologia que permite a composição de programas em uma sequência de execução com o objetivo de gerar um resultado final é chamada de *workflow* (Wf) (WFMC, 1999).

Um workflow denota a execução controlada de múltiplas tarefas em um dado ambiente. Workflows científicos são normalmente relacionados com a automação de experimentos científicos, onde programas científicos são associados, baseados em uma dependência de dados e controle (ALTINTAS, 2004). Os Sistemas de Gerenciamento de Workflows (SGWf) são *engines* de coordenação automatizados que permitem especificar, instanciar, executar, auditar e evoluir um workflow (WFMC, 1999). As principais vantagens de um WfMS são combinar uma *engine* de execução de workflow com suporte semântico para registrar e ajudar na análise da execução e re-execução do workflow (CRUZ, 2008a).

Entretanto, esse processo de concepção de workflows científicos ainda ocorre de forma ad-hoc, levando a resultados não reproduzíveis devido à ausência de métodos ou processos predefinidos para esta concepção. Em conjunto com este problema, os SGWf estão se tornando cada vez mais complexos e provendo mais funcionalidades. A cada dia mais programas e serviços estão disponíveis e descobrir como eles podem ser encadeados se torna cada vez mais difícil.

Ainda no contexto de workflows científicos, é comum que trechos de workflows se repitam dentro de um mesmo domínio. Assim, quem utiliza uma sequência de tarefas específica, A, B e C, por exemplo, normalmente utiliza uma tarefa D após essas três tarefas com certa frequência *f*.

A motivação deste trabalho, portanto, está na possibilidade de facilitar e agilizar o desenvolvimento de novos workflows através do histórico de workflows previamente desenvolvidos, de forma que o sucesso de desenvolvimentos passados ajude a melhorar sua qualidade dos workflows que serão desenvolvidos.

1.2. Objetivo

As pesquisas relacionadas a workflow podem se dividir em 4 categorias (ZHUGE, 2003): a) Extensões no modelo do workflow, ou seja, adicionar novos fatores relativos à aplicação, como restrição de tempo, agendamento de tarefa e avaliação de custo-benefício do modelo utilizado; b) Estabelecimento e atualização de padrões de modelo de workflow, ou seja, definição de uma linguagem comum para permitir interoperabilidades entre os SGWf, que é um dos objetivos da Workflow Management Coalition (WfMC) (WFMC, 1999); c) Semântica, que estuda a parte relativa a equivalência, otimização, loops e *deadlocks* em workflows; E d) desenvolvimento de aplicações de workflow, que pesquisa workflow baseados em agentes, desenvolvimento de grandes sistemas de workflows, workflow baseado em componentes e reutilização.

O desafio relacionado à reutilização de workflows científicos engloba a necessidade de se tirar vantagem da presença de workflows previamente elaborados para se compor novos workflows.

No domínio de comércio eletrônico, os sistemas de recomendação aplicam técnicas de mineração de dados consagradas para ajudar os usuários a encontrar os itens que eles gostariam de comprar, predizendo uma lista dos itens que usuário provavelmente mais gosta. Uma das técnicas que pode ser utilizada para criar essas listas se chama regra de associação. O objetivo de minerar regras de associação é encontrar todos os conjuntos de itens que frequentemente ocorrem de forma conjunta na base de dados e formar regras a partir destes conjuntos.

Este trabalho se encaixa na quarta categoria e tem como objetivo desenvolver um sistema de recomendação para workflows que permita sugerir as combinações de programas mais frequentes, ou seja, combinações que são utilizadas pela maioria dos usuários, de forma a acelerar o processo de construção de workflows e melhorar a qualidade dos workflows desenvolvidos, melhorando o desempenho e a precisão das recomendações frente às abordagens existentes. Assim como no comércio eletrônico, este trabalho adapta e utiliza a técnica de mineração de dados chamada de regras de associação para que ela possa predizer quais programas devem ser combinados durante o desenvolvimento de um workflow.

1.3. Organização

Este trabalho está organizado em 6 capítulos. Neste primeiro capítulo, de introdução, foram exibidos a motivação, o objetivo e a organização do trabalho.

No segundo capítulo são apresentadas as tecnologias envolvidas nesta dissertação, como sistemas de recomendação e workflows científico, assim como as abordagens existentes para recomendação em workflows científicos.

No terceiro capítulo, é feito um estudo mais aprofundado sobre a técnica de mineração de dados utilizada para que seja possível entender o porquê da escolha dessa técnica.

No quarto capítulo, é exibida a abordagem proposta, que consiste na construção de um sistema de recomendação para workflow utilizando a técnica de mineração de dados chamada de regras de associação, detalhada no Capítulo 3.

No quinto capítulo, é detalhado o protótipo que implementa a abordagem proposta. Nesse capítulo, também, é feita uma avaliação da abordagem, onde são avaliados requisitos funcionais como a precisão das recomendações e não funcionais como tempo de preparação dos workflows previamente desenvolvidos e quantidade de recursos computacionais utilizados.

No sexto capítulo, são apresentadas as contribuições e limitações desse trabalho, e trabalhos futuros.

Os apêndices apresentam exemplos de arquivos gerados durante a recomendação.

Capítulo 2 - Recomendação em Workflows Científicos

2.1. Introdução

A busca pelo conhecimento faz com que instituições de pesquisa procurem formas de melhorar a qualidade dos experimentos científicos e reduzir o tempo necessário para a sua execução. A adoção de técnicas que permitam atingir elevados ganhos de produtividade e qualidade na condução de experimentos científicos pode ser vista como um diferencial competitivo para essas instituições e, consequentemente, para seus países sede (MATTOSO, 2009).

Desta forma, a ciência, de um modo geral, tem feito cada vez mais uso de procedimentos computacionais com o intuito de lidar com o aumento constante dos volumes de dados e manipulações necessárias aos experimentos científicos.

Este capítulo trata de algumas dessas tecnologias que são relevantes para esta dissertação, como Sistemas de Recomendação, Workflows Científico e Sistemas de Recomendação de Workflow Científico.

2.2. Sistemas de Recomendação

A explosão no crescimento da WWW e a emergência do comércio eletrônico levaram ao desenvolvimento de sistemas de recomendação (RESNICK, 1997).

O Sistema de Recomendação é uma tecnologia de filtragem de informação personalizada usada tanto para predizer se um usuário em particular irá gostar de um determinado item (*prediction problem*) quanto para identificar o conjunto de N itens que serão de interesse de um certo usuário (*top-N recommendation problem*).

Nos últimos anos, os sistemas de recomendação vêm sendo usados em diferentes aplicações (HILL, 1995; SHARDANAND, 1995; JOSEPH A. KONSTAN, 1997; TERVEEN, 1997; SCHAFER, 1999; BEEFERMAN, 2000; KITTS, 2000;

MOBASHER, 2000): recomendando produtos que um consumidor gostaria de comprar; filmes, programas de TV ou músicas que um usuário gosta; identificando páginas na web que será de interesse; ou até sugerindo formas alternativas para procura de informação.

Várias abordagens para construir um sistema de recomendação foram desenvolvidas. Essas abordagens utilizam informação demográfica, informações históricas ou conteúdo (HILL, 1995; SHARDANAND, 1995; BALABANOVIC, 1997; JOSEPH A. KONSTAN, 1997; TERVEEN, 1997; BASU, 1998). Dentre elas, a filtragem colaborativa (FC), que é baseada em informações históricas, é a de maior sucesso e amplamente utilizada para construir sistemas de recomendação (RESNICK, 1994; KONSTAN JOSEPH A., 1997).

O termo filtragem colaborativa foi utilizado inicialmente por Goldberg (1992) para descrever um sistema de filtragem de email chamado de Tapestry. Neste sistema, cada usuário poderia escrever uma anotação sobre cada mensagem e compartilhar essas anotações com um grupo de usuários. O usuário poderia filtrar as mensagens escrevendo consultas sobre essas anotações. Apesar do Tapestry permitir que um indivíduo se beneficie de anotações feitas por outros usuários, o sistema requeria que o usuário escreva as consultas, muitas das vezes complexas. O primeiro sistema a gerar recomendações automaticamente foi o GroupLens (RESNICK, 1994; KONSTAN JOSEPH A., 1997), que provia aos usuários recomendações personalizadas nas postagens do Usenet. As recomendações para cada usuário eram obtidas identificando-se uma vizinhança de usuários similares e recomendando os artigos que esse grupo de usuários achava interessante.

Pesquisadores desenvolveram diversos algoritmos de filtragem colaborativa, que podem ser divididos em duas principais abordagens (BREESE, 1998). A primeira,

baseada no usuário (*user-based*) (RESNICK, 1994; SHARDANAND, 1995; KONSTAN JOSEPH A., 1997; BREESE, 1998; HERLOCKER, 1999; SARWAR, 2000), se baseia no fato de que cada pessoa pertence a um grupo grande de indivíduos com o mesmo comportamento. Como resultado, itens, como livros, filmes, produtos, etc, frequentemente comprados ou desejados por vários membros do grupo podem ser usados para formar uma base de itens recomendados. A segunda, baseada em modelo (*model-based*) (SHARDANAND, 1995; BILLSUS, 1998; BREESE, 1998; AGGARWAL, 1999; KITTS, 2000), analisa informações históricas para identificar relações entre diferentes itens tal que a compra de um item (ou um conjunto deles) geralmente implica na compra de outro item (ou outro conjunto de itens), e então usa essa relação para determinar os itens recomendados. Os esquemas baseados em modelo produzem uma recomendação rapidamente, porém, é necessário muito tempo para construir esses modelos. Além disso, essas recomendações geralmente são de pior qualidade se comparada ao esquema baseado no usuário. Em contraste, os esquemas baseados em usuário tendem a produzir sistemas que levam a recomendações de alta qualidade, porém, sofrem de vários problemas de escalabilidade (MUKUND, 2004).

2.3. Workflow Científico

Atualmente, diversos pesquisadores desenvolvem seus experimentos científicos utilizando composição de programas, onde cada programa produz um conjunto de dados com determinada semântica e sintaxe. Esse conjunto poderá ser utilizado como entrada de dados para o próximo programa (CRUZ, 2008b).

Desconsiderando a importância e a diferença entre os experimentos, eles têm uma abordagem comum, permitindo pesquisadores analisarem grandes quantidades de dados através de processos semi-automáticos construídos por vários programas e banco de dados. Porém, essa abordagem tem uma grande desvantagem, pois utiliza *pipelines*

(sequência de programas conectados, onde a saída de um programa é a entrada de outro) baseados em *scripts*, que são fortemente ligados ao código fonte e de difícil gerenciamento. Na computação, a tecnologia que permite a composição de programas em uma sequência de execução com o objetivo de gerar um resultado final é chamada de *workflow* (Wf).

A definição de workflow, segundo a WfMC (WFMC, 1999) é:

"A automação de um processo de negócio, completo ou apenas parte dele, através do qual, documentos, informações ou tarefas são transmitidos de um participante a outro por ações, de acordo com regras procedimentais."

Contudo, o cenário atual nos remete aos primórdios da computação, pois centros renomados de pesquisa ainda dependem exclusivamente da capacidade individual dos cientistas no encadeamento dos programas necessários para a execução de experimentos. Esse cenário é sujeito a falhas e improdutivo, especialmente em se tratando de experimentos complexos, que podem envolver muitos programas, grande quantidade de dados e diversos cientistas em localidades geograficamente dispersas.

Com o intuito de atenuar esse cenário, sistemas de gerência de workflows científicos (SGWfC) passaram a ser utilizados. Os SGWfCs melhoram a reutilização de componentes de workflow e promovem o compartilhamento do conhecimento em diferentes domínios. Entretanto, a diversidade de domínios de aplicações científicas e a diversidade de cenários de utilização em e-Science resulta no aparecimento de diversos SGWfC, que em alguns casos, são direcionados a um domínio científico específico. Alguns dos SGWfC mais conhecidos são: Taverna (HULL, 2006), Kepler (ALTINTAS, 2004) , VisTrails (CALLAHAN, 2006), Swift (ZHAO, 2005) e OMII-BPEL (TAYLOR, 2006).

Lemos (2004) exibe uma lista de requisitos que os SGWfC devem atender. Destacamos estes requisitos a seguir:

O SGWfC deve incluir os **processos, dados e recursos** normalmente usados e permitir sua extensão com novos processos, dados e recursos. O SGWfC também deve auxiliar os pesquisadores na **definição e redefinição** do workflow. Quando os resultados não forem úteis ou relevantes para os pesquisadores, é muito importante que o workflow possa ser redefinido. Para isto, o sistema deve permitir a definição de propriedades dos processos, dados e recursos de tal forma a facilitar a escolha dos mais adequados para cada situação ou problema. Devem ser oferecidas ferramentas para **validar** o workflow definido pelo cientista. Durante a validação, o sistema deve verificar se as entradas e saídas definidas pelos pesquisadores para cada processo do workflow são consistentes, além de incluir, caso seja necessário, processos que façam conversão nos formatos dos dados e processos que verifiquem se os resultados gerados pelos programas são esperados ou não. O SGWfC deve ser capaz de **otimizar** e **executar** o workflow definido pelo pesquisador. A execução do workflow pode ser **monitorada** e deve permitir a intervenção do pesquisador em qualquer ponto. A intervenção (interrupção temporária) é necessária caso ele queira avaliar os resultados intermediários para decidir se continua ou não a execução, ou para fazer alguma modificação na definição das próximas atividades do workflow. O sistema deve oferecer **agendamento** da execução do workflow, ou seja, deve permitir que o pesquisador execute o workflow uma única vez em um determinado dia ou de tempos em tempos (diariamente, semanalmente, etc). O SGWfC deve armazenar tanto os **dados** produzidos pelo workflow quanto os **metadados**. Metadado é comumente definido como "dado sobre dado", ou ainda, de forma mais completa, como uma informação sobre o dado que permite o acesso e gerenciamento deste dado de maneira eficiente e

inteligente (SUMPTER, 1994). O sistema deve ser capaz de gerar estes metadados automaticamente e, sempre que possível, oferecer subsídios para que o cientista consulte-os e atualize-os.

Um exemplo de SGWfC pode ser visto na Figura 2-1. Este sistema se chama VisTrails. À esquerda da figura, estão as tarefas que podem ser adicionadas ao workflow. No meio, está o workflow propriamente dito, com as tarefas encadeadas de acordo com as dependências. À direita, estão as propriedades de cada tarefa.

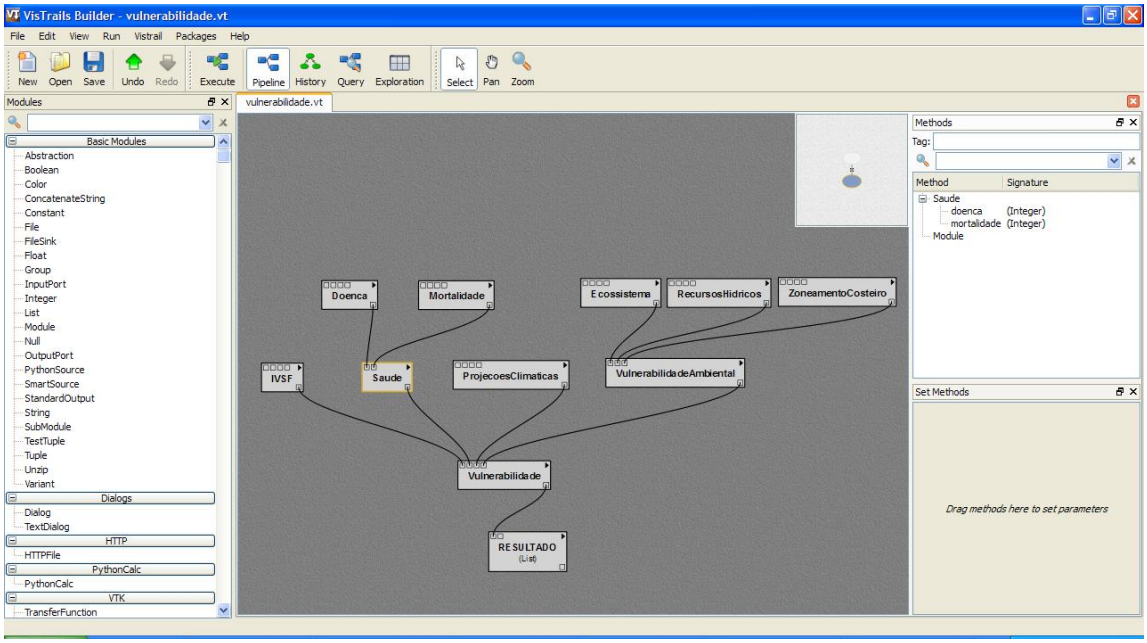


Figura 2-1 – Interface do VisTrails

2.4. Sistemas de Recomendação em Workflows Científicos

O desafio relacionado à reutilização de workflows científicos engloba a necessidade de se tirar vantagem da presença de workflows previamente elaborados para se compor novos workflows.

Os SGWfC estão se tornando cada vez mais complicados de serem usados e provendo mais funcionalidades. A cada dia, mais serviços e tarefas estão disponíveis e a combinação entre eles se torna mais complexa.

O encadeamento de atividades num workflow não é uma tarefa trivial e, em muitos casos, se transforma em uma barreira para a construção de análises e modelos mais sofisticados. SGWfC, como Taverna, Kepler e VisTrails, oferecem uma interface gráfica rica a partir da qual componentes previamente registrados podem ser arrastados ou colocados na área de edição de workflow. A busca por atividades é limitada, visto que, por exemplo, não há registro de atividades equivalentes no SGWfC. O conhecimento de quais atividades podem ser interligadas é ainda tácito. É necessário estudar um grande número de exemplos de workflows para se ganhar experiência na configuração do fluxo das atividades (MATTOSO, 2009).

Com o intuito de minimizar esses problemas, surgiram sistemas de recomendação de workflow científicos, como o SmartLink (TELEA, 2000) e VisComplete (KOOP, 2008). A abordagem que deu origem a esta dissertação pode ser encontrada em Oliveira (2008).

O **SmartLink** surgiu para minimizar os problemas existentes na programação visual (*visual programmable dataflow system*), que tem por objetivo permitir a criação de novos experimentos para processamento e visualização de dados 3D, extração de imagens e padrões.

A programação visual permite a criação de programas a partir de módulos que representam componentes de códigos, onde estes módulos são encadeados para formar estes programas. Porém, tais sistemas apresentam bibliotecas enormes, contendo centenas de módulos.

Alguns exemplos de ambiente de programação visual são: AVS ou Express (UPSON, 1989), Iris Explorer ou Khoros (RASURE, 1994) e Vistrails; e alguns exemplos de bibliotecas de programação são VTK (SCHROEDER, 1995) e Java3D (SOWIZRAL, 1998).

O SmartLink foi implementado inicialmente para o sistema de visualização Vission (TELEA, 1999). Alguns termos utilizados pelo Smartlink são definidos a seguir: Um *grafo* consiste de *nós*, que representam as portas dos módulos, e *links*, que representam o fluxo de dados entre essas portas; um *caminho* é uma sequência de *links* no grafo de uma porta até outra. O SmartLink mantém um banco de dados capaz de armazenar um conjunto de *links* para todas as portas dos módulos existentes no sistema, formando um grafo de preferências. Esses *links* representam as conexões mais utilizadas no sistema. Para cada *link*, um número inteiro é armazenado para representar o número de vezes que esta conexão ocorreu.

As informações armazenadas no banco de dados são então utilizada para auxiliar na construção de novos aplicativos. A proposta do SmartLink é auxiliar o cientista, respondendo às seguintes perguntas: "Como posso conectar dois módulos?" e "O que pode ser conectado ao módulo?"

Para ilustrar, considere a Figura 2-2, onde o usuário sabe previamente que utilizará os módulos VTKQuadric, VTKContourFilter e VTKViewer, e sabe também que a porta de entrada utilizada pelo VTKContourFilter é a porta 2. Porém, ele não sabe o que há entre os módulos VTKQuadric e VTKContourFilter.

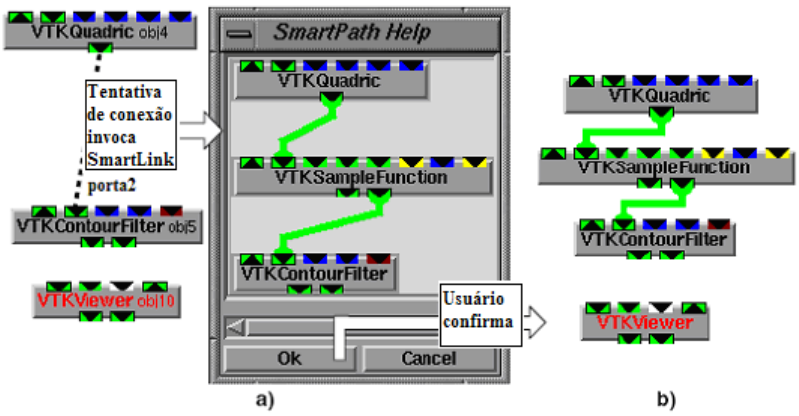


Figura 2-2 - Usando SmartLink para conectar os módulos VTKQuadric e VTKContourFilter. ***Fonte: (TELEA, 2000)***

Ao conectar os módulos, o SmartLink inicia uma busca em profundidade (*breadth-first*) no conjunto de grafos de preferência considerando a saída do VTKQuadric e a entrada (porta2) do VTKContourFilter. Neste caso, o sistema encontra um caminho passando pelo módulo VTKSampleFunction e o recomenda ao usuário (Figura 2-2.a). Caso o usuário aceite a recomendação, o sistema automaticamente preenche o workflows com os módulos necessários (Figura 2-2.b). Esta situação ilustra a primeira pergunta descrita anteriormente.

Para ilustrar a segunda pergunta ("O que pode ser conectado ao módulo?"), considere que se deseja obter sugestões de módulos que podem ser conectados a uma determinada porta. Na Figura 2-3, deseja-se saber qual módulo pode ser conectado à porta "propriedade" do módulo VTKActor (Figura 2-3.a). No caso, o módulo VTKProperty é sugerido (Figura 2-3.b) e conectado ao workflow (Figura 2-3.c). Este tipo de construção também é conhecido como construção em cascata.

De acordo com o autor, os usuários acharam o SmartLink bastante efetivo. Os usuários experientes se beneficiaram mais do método de construção em cascata, aumentando a velocidade de construção dos workflows consideravelmente. Para os usuários novos, é necessário obter uma cópia do banco de um expert de seu domínio para começar a usufruir d as recomendações tão logo iniciem a utilização do sistema.

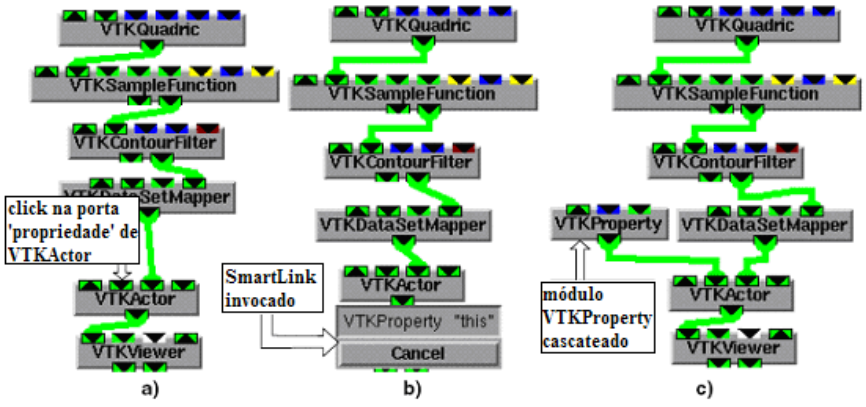


Figura 2-3 - a) Porta "propriedade" do VTKActor selecionada. b) Popup menu com as opções de módulos. c) Modulo VTKProperty conectado. ***Fonte: (TELEA, 2000)***

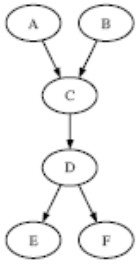
O **VisComplete** foi apresentado como um sistema para auxiliar os usuários no processo de criação de visualizações usando um banco de dados (coleção) de pipelines previamente criados.

O VisComplete modela os pipelines como grafos, onde os nós são representados pelos módulos (tarefas) e os vértices determinam como os dados fluem entre os módulos. Desta forma, foi possível desenvolver um algoritmo para sugerir trechos de pipelines, oriundos de buscas em subgrafos comuns na coleção, que pudessem completar o pipeline em desenvolvimento.

O problema de completar pipelines foi definido da seguinte forma: Dado um grafo parcial G, deseja-se encontrar um conjunto de preenchimentos (*completion*) C(G) que reflita as estruturas existentes em uma coleção de grafos completos (finalizados). Um preenchimento (*completion*) de G, Gc, é um supergrafo de G.

A sua solução consiste de dois passos. Primeiro, é feito um preprocessamento na coleção de pipelines P e cria-se um P_{path}, uma representação compacta de P que sumariza os relacionamentos entre estruturas comuns (ex. sequências de módulos) na coleção. Em seguida, dado um pipeline parcial *p*, complementos são gerados a partir da análise do P_{path} para se identificar módulos e conexões que já foram usados em conjunto com *p* na coleção.

A Figura 2-4 ilustra como é feita a criação do P_{path} dado um pipeline completo.



Caminho	Vértice
$A \rightarrow C \rightarrow D$	E
$A \rightarrow C \rightarrow D$	F
$B \rightarrow C \rightarrow D$	E
$B \rightarrow C \rightarrow D$	F
$C \rightarrow D$	E
$C \rightarrow D$	F
D	E
D	F

Figura 2-4 - Criação dos caminhos para um determinado pipeline. *Fonte: (KOOP, 2008).*

A recomendação dos complementos no Viscomplete se dá de forma iterativa, tanto no sentido bottom-up do pipeline, quanto no top-down. A Figura 2-5 ilustra esta recomendação.

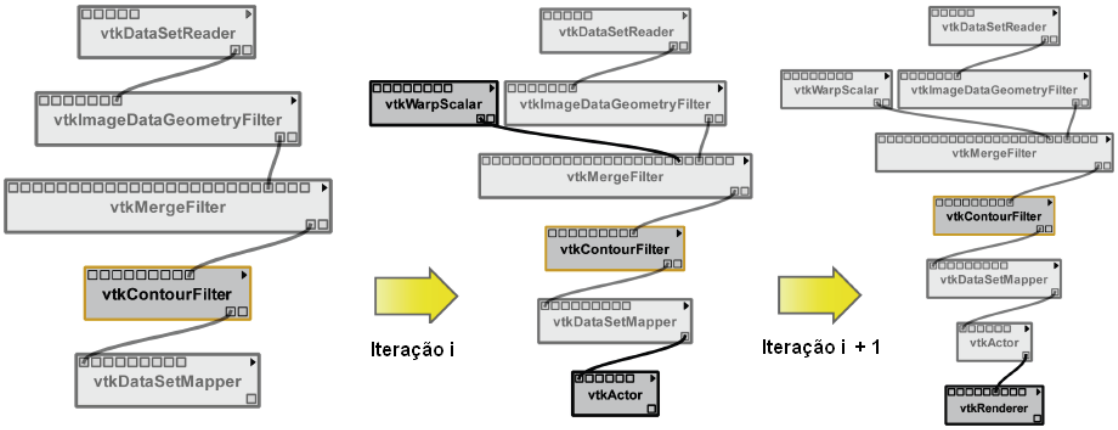


Figura 2-5 - As predições são refinadas iterativamente. ***Fonte: (KOOP, 2008).***

A cada passo, a predição pode ser sugerida tanto para a entrada quanto para a saída do módulo, ou seja, em ambas as direções; na segunda iteração, o algoritmo sugere apenas para a saída do módulo. As predições em ambas as direções podem gerar ramos no pipeline, como apresentado na figura do meio.

Para cada caminho existente no pipeline, o VisComplete calcula as possíveis recomendações e analisa qual deve ser exibida ao usuário. A Figura 2-6.b exibe os dois possíveis caminhos do fragmento de pipeline da Figura 2-6.a e suas duas possíveis recomendação com as respectivas ocorrências. Como há um número maior de ocorrências para o módulo vtkDataSetMapper, este seria sugerido em primeiro plano.

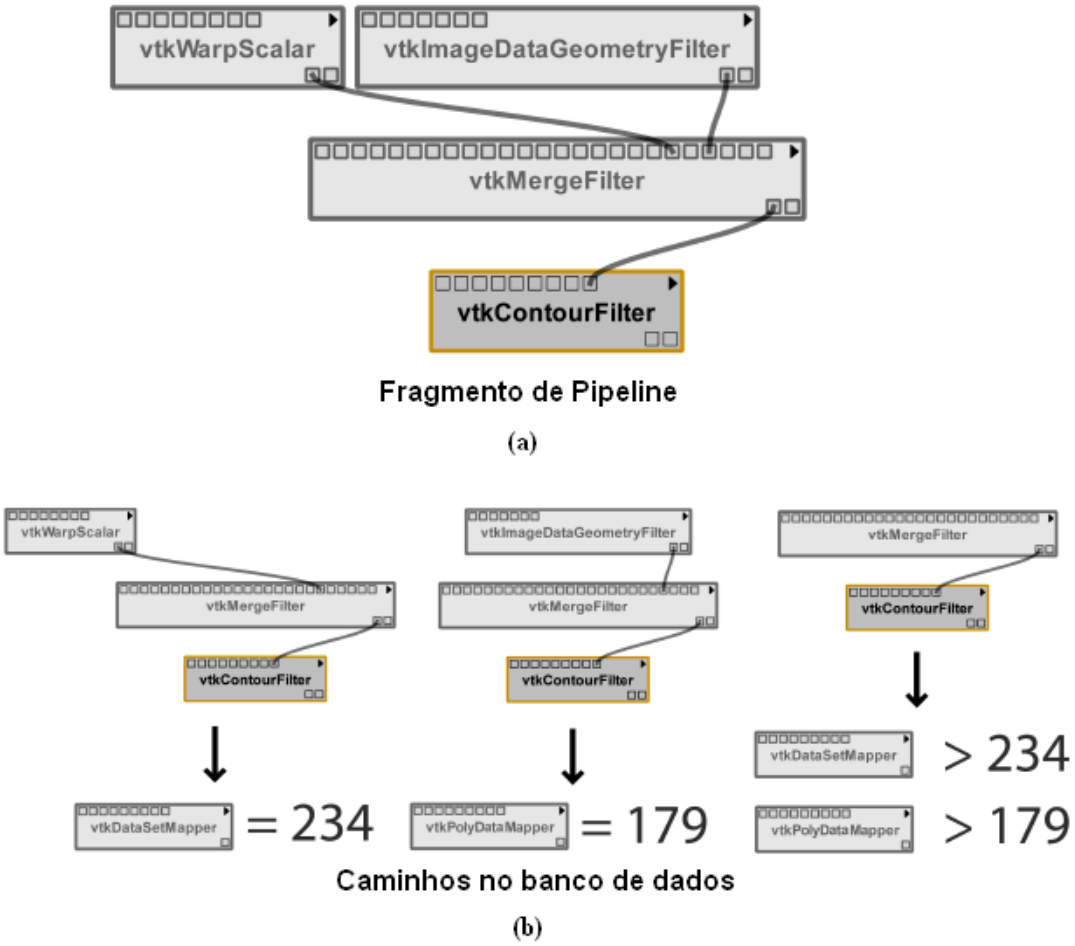


Figura 2-6 - Análise de todos os caminhos para gerar uma recomendação. *Fonte: (KOOP, 2008).*

De acordo com o autor, estudos preliminares mostraram que o VisComplete tem o potencial de reduzir o esforço e tempo empregado na construção de visualizações (KOOP, 2008).

A abordagem proposta em Oliveira (2008) foi inicialmente concebida para permitir uma recomendação pró-ativa de forma incremental, onde para cada tarefa selecionada, apenas uma nova tarefa é sugerida.

Inicialmente, as relações entre as tarefas são extraídas. Cada uma dessas relações é composta pela tarefa origem e pela tarefa destino, além de informações de como elas são conectadas, ou seja, por quais portas. Assim, à medida que um usuário adiciona uma nova tarefa ao workflow em desenvolvimento, o sistema, de forma pró-ativa, verifica

qual a tarefa mais relevante que pode ser conectada à tarefa inserida anteriormente, indicando como eles podem ser conectados.

A Figura 2-7 mostra a implementação desta abordagem para o VisTrails, onde ao se adicionar a tarefa HmmBuild, o sistema mostra as quatro possíveis tarefas e formas de conexão que podem suceder a tarefa adicionada. Por exemplo, 40% das tarefas que sucedem HmmBuild são conectadas pela porta StdOut à porta HmmPath da tarefa HmmCalibrate.

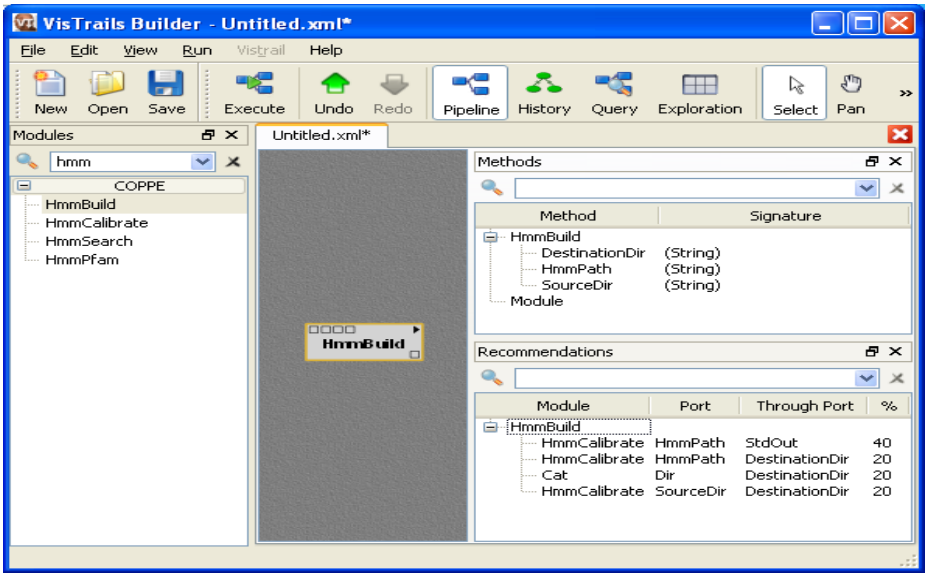


Figura 2-7 - Abordagem proposta por Oliveira (2008)

Ambos os sistemas descritos nesta seção contribuem para melhorar o desenvolvimento de novos workflows, porém eles apresentam alguns problemas que serão expostos a seguir.

Um primeiro ponto a observar sobre o SmartLink é que sua recomendação é baseada nas portas dos módulos. Como dito anteriormente, se a escolha e entendimento dos módulos é uma tarefa difícil, a escolha de uma porta como base da recomendação é ainda mais difícil. Assim, para um melhor aproveitamento do sistema, o pesquisador precisará de um conhecimento significativo dos módulos e suas portas.

Outro ponto negativo do SmartLink é que o sistema não considera os módulos previamente inseridos no workflow. O sistema apenas considera a porta origem e a porta destino para efetuar o cálculo para recomendação. Considere a Figura 2-8.a, onde se deseja conectar a saída 2 de VTKContourFilter (origem) à entrada 1 de VTKViewer (destino). Os resultados são apresentados na Figura 2-8.b, em ordem de relevância, da esquerda para direita. Para ilustrar uma consequência negativa desta abordagem, considere que o módulo VTKSampleFunction tenha uma forte ligação com o módulo VTKAssembly. Ao se desconsiderar os módulos previamente inseridos, VTKQuadric e VTKSampleFunction, a recomendação que contém o módulo VTKAssembly fica em último lugar, pois é desconsiderada a sinergia entre os módulos VTKSampleFunction e VTKAssembly, fato que não deveria ser descartado.

Ainda em relação à Figura 2-8, ao considerar somente a porta origem e destino, o sistema também limita a quantidade de módulos recomendados, pois o sistema poderia recomendar outros módulos após o VTKViewer, como por exemplo, VTKCamera, diminuindo um pouco sua eficiência.

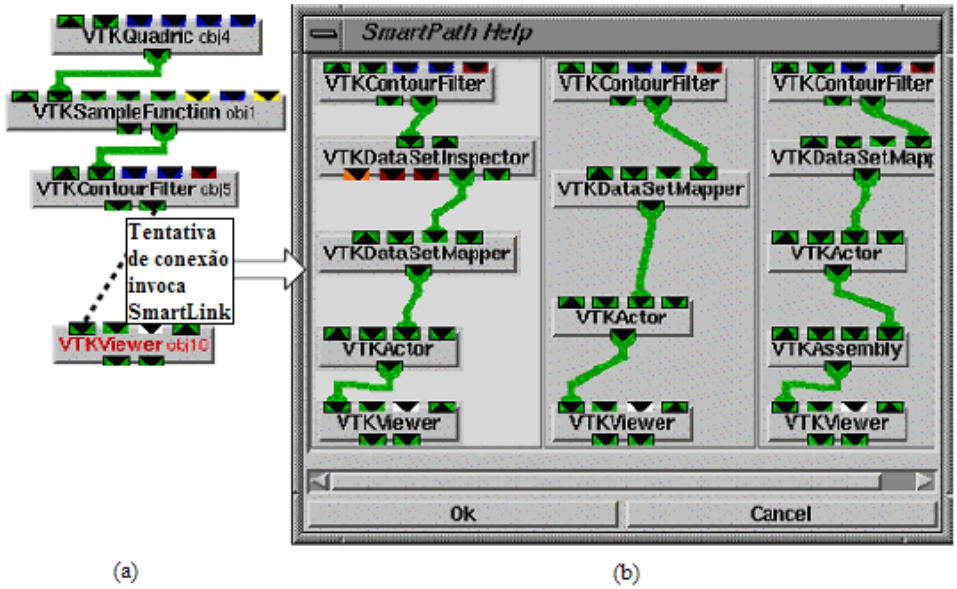


Figura 2-8 - Exibição de múltiplas recomendações no SmartLink. *Fonte:* (TELEA, 2000).

O VisComplete apresenta uma abordagem mais genérica e abrangente, pois reduz os workflows a um conjunto de caminhos e gera recomendações se baseando tanto numa abordagem bottom-up quanto top-down. Porém, como a geração dos caminhos não se baseia em nenhuma técnica de mineração de dados, o VisComplete falha na recomendação de workflows que contém ruídos, ou módulos pouco usados.

Os ruídos são módulos que o usuário deseja utilizar, mas são pouco, ou até nunca, utilizados. Este módulo ruído, na visão deste trabalho, não altera a finalidade principal do workflow em desenvolvimento, podendo ser descartado na busca por uma recomendação para o workflow como um todo.

Para melhor ilustrar, considere a seguinte sequência de módulos que está armazenada na base de dados de pipelines desenvolvidos: A→B→C→D; e considere a seguinte sequência em desenvolvimento: A→X→B, onde X é um módulo pouco utilizado e não se encontra em nenhuma sequência A→... →B→ ... →C da base. Se for feita uma comparação diretamente com a base, como é feito no VisComplete, a sequencia A→X→B não aparece nenhuma vez, não gerando recomendação. Tal problema poderia ser resolvido com a utilização de técnicas de mineração de dados, como por exemplo, mineração sequencial. Soluções para resolver tal problema serão descritas nos próximos capítulos.

A abordagem proposta por Oliveira (2008) não considera as tarefas previamente inseridas, resultando no mesmo problema presente no smartlink.

2.5. Considerações Finais

Neste capítulo, foi apresentado o que são sistemas de recomendação e para que servem. Foi exposta a importância dos workflows científicos, assim como as características desejáveis de um sistema de gerenciamento de workflow científico. Foi

descrito também um pouco da dificuldade de se utilizar os SGWfC, o que motivou a criação dos sistemas de recomendação para workflows científicos.

Dois sistemas de recomendação de workflows científicos foram analisados e com base nas suas qualidades e nas suas falhas, foi proposto um sistema de recomendação em workflows científicos novo que está descrito no Capítulo 4.

O próximo capítulo aborda técnicas de regras de associação, mais precisamente, técnicas de mineração sequencial, assim como os algoritmos que as implementam. Esta técnica é essencial para tratar problemas de ruído, conforme exposto neste capítulo.

Capítulo 3 - Descoberta do Conhecimento em Base de Dados

3.1. Introdução

Fayyad (1996a) apresenta a definição clássica do processo de descoberta de conhecimento em bases de dados, conhecido por KDD (*Knowledge Discovery in Databases)*:

"KDD é um processo, de várias etapas, não trivial, interativo e iterativo, para identificação de padrões compreensíveis, válidos, novos e potencialmente úteis a partir de grandes conjuntos de dados".

As técnicas de KDD (FAYYAD, 1996a), também conhecidas como mineração de dados, normalmente se referem à extração de informações implícitas, porém úteis, de um banco de dados. As aplicações de KDD tipicamente envolvem o uso de mineração de dados para descobrir um novo modelo, e então os analistas aplicam esse modelo em suas aplicações.

Os objetivos do processo de KDD podem ser divididos em dois tipos: verificação e descoberta (FAYYAD, 1996a). Na verificação, o sistema se limita a verificar hipóteses estabelecidas pelo usuário. Na descoberta, o sistema autonomamente descobre novos padrões ocultos nos dados. Este último pode ser subdividido em descoberta preditiva, onde o sistema descobre padrões para predizer o comportamento futuro de algumas entidades, e descoberta descritiva, onde o sistema encontra padrões a serem apresentados de uma forma inteligível aos seres humanos. Neste trabalho, o tipo de KDD utilizado será a descoberta preditiva, onde se procura predizer como as tarefas de um workflow podem ser encadeadas.

O processo de Descoberta de Conhecimento em Bases de Dados é basicamente composto por três grandes etapas: Pré-processamento, Mineração de Dados e Pós-

processamento. A mostra todo o processo de KDD. A seguir, é discutido um pouco sobre cada etapa.

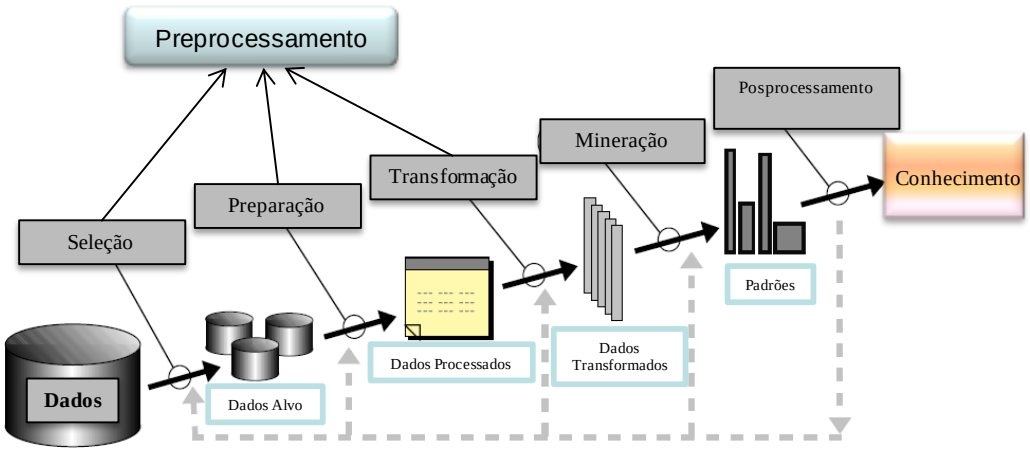


Figura 3-1 - Visão geral dos passos que compõem o processo de KDD. ***Fonte: (FAYYAD, 1996b).***

A primeira etapa do processo de KDD, conhecida como **pré-processamento**, é a etapa responsável por selecionar, preparar e transformar os dados que serão utilizados pelo processo. Vale ressaltar, que dados com erros, incompletos ou redundantes podem comprometer todo o processo.

A segunda etapa do KDD, conhecida como **Mineração de Dados**, é a aplicação de um algoritmo específico para extrair padrões de dados.

Hand (2001) define a etapa de mineração de dados da seguinte forma: "*Mineração de Dados é a análise de (quase sempre grandes) conjuntos dados observados para descobrir relações escondidas e para consolidar os dados de uma forma tal que eles sejam inteligíveis e úteis aos seus donos*".

A etapa de Mineração de Dados normalmente é a que atrai maior atenção, por ser ela que revela os padrões ocultos nos dados. Algumas tarefas que podem ser realizadas durante a etapa de Mineração de Dados são descritas a seguir:

A descoberta de **regras de associação** (AGRAWAL, 1993; SARWAR, 2000). O principal objetivo é encontrar regras entre dois conjuntos de produtos no banco de dados das transações (um exemplo de transação seria uma compra efetuada no supermercado), ou seja, a presença de um produto em um conjunto implica a presença de outros produtos de um outro conjunto. Apriori (AGRAWAL, 1994), DHP (PARK, 1995), Tree Projection (AGARWAL, 2000) e FP-tree (JIAWEI, 2000) são alguns dos algoritmos mais conhecidos para encontrar regras de associação em um banco de dados. Este assunto será abordado com mais detalhes neste capítulo, tendo em vista sua relevância para esta dissertação.

A tarefa de **classificação** tem como objetivo descobrir uma função capaz de mapear (classificar) um item em uma de várias classes pre-definidas (HAND, 1981; WEISS 1991). Se conseguirmos obter a função que realiza este mapeamento, quaisquer novas tuplas podem ser também mapeadas, sem a necessidade de conhecimento prévio da sua classe.

A tarefa de **agrupamento** reúne, em um mesmo grupo, objetos de uma coleção que mantenham algum grau de afinidade. Assim, a sua base é o conceito de similaridade, e o seu objetivo principal é o de maximizar a similaridade de objetos do mesmo grupo, e de minimizá-la entre os elementos de grupos (JAIN, 1999).

A diferença principal entre esta tarefa e a de classificação é que, na classificação, conhecemos previamente a que classe os elementos de uma coleção pertencem.

A **regressão** linear analisa a relação entre duas variáveis, X e Y, onde se deseja encontrar a melhor reta que passa por esse pontos. Assim, podemos usar a reta descoberta para encontrar, ou predizer, novos valores de X a partir de Y ou novos valores de Y a partir de X.

A última etapa do KDD, o **Pós-Processamento**, tem como objetivo transformar os padrões dos dados obtidos na etapa anterior, de forma a torná-los inteligíveis, tanto ao analista de dados quanto ao especialista do domínio da aplicação (SOARES, 2007).

3.2. Regras de Associação

A descoberta de regras de associação é uma das tarefas fundamentais da mineração de dados. Ela tem como objetivo procurar por dependências em uma vasta quantidade de dados. Esta tarefa resulta na chamada regra de associação, que é da forma: Se *A* ocorre nesse conjunto de dados então *B* também ocorre. Somente as regras que ocorrem com frequência suficiente é que são geradas e exibidas ao usuário.

As Regras de Associação, essencialmente, se preocupam em descobrir associações entre dois conjuntos de produtos tal que a presença de alguns produtos em uma transação em particular implica que produtos do outro conjunto também estejam presentes na mesma transação (AGRAWAL, 1993). Existem dois indicadores que determinam a qualidade da regra. Esses indicadores são descritos mais adiante.

Mais formalmente, seja *I* um conjunto de m itens $\{I_1, I_2, ..., I_m\}$. Seja *T*, um conjunto de transações, onde cada transação *t* é representada por t[k]=1, se *t* comprou o item I_k, e t[k]=0, caso contrário. Por regra de associação, entendemos como uma implicação da forma X=>Y, onde X, antecedente, e Y, conseqüente, representam um conjunto de itens de I e X∩Y=Ø. Dizemos que a regra X=>Y com um fator de confiança c (0 < c < 1) se c% das transações de T que satisfazem X também satisfazem Y. Ou seja, a confiança é a probabilidade condicional de uma regra, onde cada transação *t* que contém X, *t* também contem Y. Para a regra *R*, X=>Y, temos:

$$\text{confiança}(R) = P(Y|X) = \frac{P(Y \wedge X)}{P(X)}$$

A confiança reflete a validade de uma regra, procura expressar a qualidade dela, indicando o quanto a ocorrência do seu antecedente pode assegurar a ocorrência do seu consequente.

Um outro fator para determinar a qualidade da regra é o suporte, que representa uma fração de transações em T que satisfazem a união dos itens dos antecedentes com os consequentes, de forma simplificada, é a frequência com que uma regra ocorre dentro de um conjunto de regras (AGRAWAL, 1993):

$$s = \frac{\text{número de transações contendo X U Y}}{\text{número de transações}}$$

A confiança de uma regra revela o quanto esta regra é aplicada, enquanto que o suporte indica o quanto esta regra é confiável. Portanto, para uma regra ser relevante é preciso ter bastante suporte e suficiente confiança. Assim pode-se dizer que uma regra é relevante ou forte se esta apresenta suporte e confiança acima de valores prefixados de confiança mínima e de suporte mínimo.

Uma utilização prática das regras de associação é o sistema de recomendação que retorna os *N* produtos mais relevantes para um usuário (*top-N recommendation problem*). Esses sistemas funcionam da seguinte maneira. Um banco de dados é alimentado com transações de consumidores. Cada consumidor terá apenas uma transação contendo todos os itens comprados por ele até o momento. Para um conjunto de *n* consumidores serão criadas *n* transações. Em seguida, é utilizado um algoritmo de descoberta de regra de associação para encontrar todas as regras que satisfazem o suporte e confiança mínimo estabelecidos no sistema. Agora, para cada consumidor *u* que gostaríamos de encontrar seus *N* produtos prediletos, três passos são executados. O primeiro passo consiste em encontrar todas as regras que são suportadas pelo consumidor, ou seja, encontrar todas as regras cujo antecedente (lado esquerdo da regra)

contenha produtos já comprados por *u*. Seja P_u o conjunto de produtos únicos que estão sendo preditos por todas essas regras e ainda não foram comprados pelo consumidor u. O segundo passo consiste em ordenar os produtos contidos em P_u de acordo com a confiança da regra que continham esses produtos como consequentes (lado direito da regra), para que produtos preditos por regras que contém alta confiança sejam classificas primeiro. Finalmente, são selecionados os primeiros N produtos melhor classificados para compor o conjunto de recomendações (SARWAR, 2000).

Em Goldschmidt (2005), encontramos um exemplo didático de como obter associações a partir de um conjunto de dados. Este exemplo, apresentado na Tabela 3.21, contém transações de um supermercado. Cada transação é uma operação de venda e o objetivo é descobrir produtos que sejam vendidos de forma conjunta frequentemente.

Tabela 3.21- Relação das vendas de um Mercado durante um período de tempo

Transação	**Leite**	**Café**	**Cerveja**	**Pão**	**Manteiga**	**Arroz**	**Feijão**
1	Não	Sim	Não	Sim	Sim	Não	Não
2	Sim	Não	Sim	Sim	Sim	Não	Não
3	Não	Sim	Não	Sim	Sim	Não	Não
4	Sim	Sim	Não	Sim	Sim	Não	Não
5	Não	Não	Sim	Não	Não	Não	Não
6	Não	Não	Não	Não	Sim	Não	Não
7	Não	Não	Não	Sim	Não	Não	Não
8	Não	Não	Não	Não	Não	Não	Sim
9	Não	Não	Não	Não	Não	Sim	Sim
10	Não	Não	Não	Não	Não	Sim	Não

Observando a Tabela 3.21, podemos deduzir algumas regras, com os respectivos suporte (s) e confiança (c):

1 - *Leite* → *Pão* s. 20% , c.100%

2 - *Café* → *Pão* s. 30%, c. 100%

3 - *Pão* e *Manteiga* → *Café* s. 30%, c. 75%

4 - *Café* e *Pão* → *Manteiga* s. 30%, c. 100%

Para o item 1, entende-se que 20% das pessoas que compram algum produto neste supermercado, compraram leite e pão. E 100% das pessoas que compraram leite, compraram pão. Para o item 3, 30% das pessoas que compraram algum produto neste supermercado compraram pão, manteiga e café. E 75% das pessoas que compraram pão e manteiga compraram café.

Existem diversos algoritmos desenvolvidos especificamente para aplicação na tarefa de descoberta de associações. O mais famoso é o *Apriori* (AGRAWAL, 1993). Porém, existem outras opções, tais como DHP–*Direct Hashing and Pruning* (HOLT, 1999), DIC–*Dynamic Itemset Counting* (BRIN, 1997), *Eclat* e *MaxEclat* (ZAKI, 1997) e *EstMerge* (SRIKANT, AGRAWAL, 1997) e FP-Growth (JIAWEI, 2000).

Entretanto, várias fontes de informação geram dados com uma natureza sequencial inerente, i.e. compostos por eventos discretos que tem uma ordem temporal ou espacial. Este tipo de dado pode ser obtido, por exemplo, através de redes de telecomunicações, comércio eletrônico e banco de dados genéticos. Para este tipo de dado, a descoberta de regra de associação não é suficiente, e em meados dos anos 90, inicialmente introduzido por Agrawal (1995), surgiu o interesse por mineração sequencial dentro da comunidade de mineração de dados. Os padrões sequenciais podem ser representados da forma: quando A ocorre, B também irá ocorrer em algum momento.

Tendo em vista que os workflows são grafos orientados e a descoberta de regras de associação não considera a ordem cronológica que os produtos de uma transação foram comprados, a recomendação em workflow também se encaixa nos casos expostos no parágrafo anterior, onde é preciso utilizar uma técnica para gerar regras de associação que considere a ordem dos elementos. Esta técnica será descrita na próxima seção.

3.3.Mineração Sequencial

Em 1995, Agrawal introduziu o problema de mineração de padrões sequenciais sobre uma base de dados. A novidade na descoberta de padrões sequenciais, se comparada ao método tradicional de descoberta de regras de associação, está na inclusão de dados temporais tanto nas regras quanto no processo de mineração. Um exemplo de tal padrão é que, normalmente, um cliente aluga primeiro o filme Harry Potter e a Pedra Filosofal e depois aluga Harry Potter e a Câmara Secreta. Note que esses aluguéis não precisam ser consecutivos. Ou seja, os clientes que alugarem outros filmes entre esses dois também atenderão a este padrão sequencial.

Os elementos de uma sequência não precisam ser itens simples. Por exemplo, uma compra de travesseiro e colchão, normalmente é seguida de fronha e lençol. Neste exemplo, cada elemento deste padrão sequencial é composto por um conjunto de itens.

Os dados em sequência são caracterizados por três atributos: objeto, *timestamp* e evento (JOSHI, 2001); que podem ser mapeados para o domínio de comércio eletrônico como: identificação do cliente, data da transação e itens comprados em uma transação. É considerado também que nenhum cliente pode ter mais de uma transação com o mesmo *timestamp*.

Agrawal (1995) definiu o problema de mineração sequencial da seguinte maneira: O “conjunto de itens” é um conjunto não vazio. Uma “sequência” é uma lista

ordenada de “conjunto de itens”. O “conjunto de itens” *c* é representado por $\{i_1\ i_2\ i_3\ ...\ i_n\}$, onde i_j é um inteiro que representa um item. A sequência *s* é representada por $< c_1\ c_2\ c_3\ ...\ c_n >$, onde c_j é um conjunto de itens.

Uma sequência $< a_1\ a_2\ a_3 ... a_n >$ está contida em outra sequência $< b_1\ b_2\ b_3 ... b_n >$ se existem inteiros $i_1 < i_2 < i_3 ... < i_n$ tais que $a_1 \subset b_{i1}$, $a_2 \subset b_{i2}$, ... , $a_n \subset b_{in}$. Por exemplo, a sequência < (3) (4 5) (8) > $\subset$ < (7) (3 8) (4 5 6) (8) > pois (3) $\subset$ (3 8) , (4 5) $\subset$ (4 5 6) e (8) $\subset$ (8). Em um conjunto de sequências, a sequência *s* é máxima se *s* não está contida em nenhuma outra sequência.

Todas as transações de um cliente podem ser vistas como uma sequência, onde cada transação corresponde a um conjunto de itens. Esta sequência, sequência-cliente, está ordenada pela data da transação de forma crescente. Um cliente “suporta” uma sequência *s* se *s* está contida nas sequências-cliente deste cliente. O suporte de uma sequência é a fração total de consumidores que “suportam” essa sequência.

Dado um banco de dados *BD* de transações de clientes, o problema de minerar padrões sequenciais é encontrar as sequências máximas dentre todas as sequências que atingiram o suporte mínimo determinado. Cada uma dessas sequências máximas será um *padrão sequencial*.

Para melhor ilustrar, considere a Figura 3.2. A Figura 3.2 a ilustra uma base de dados, ordenada pela identificação do cliente (Id Cliente) e pela data da transação (*timestamp*). A Figura 3.2b exibe a base de dados como um conjunto de sequências de itens comprados por cliente (sequência-cliente). A Figura 3.2c exibe o resultado para um suporte de sequência mínimo de 25%. Para um suporte mínimo de 25%, i.e., um suporte mínimo de dois consumidores, tem-se duas sequências: < (30) (90) > e < (30) (40,70) >. A sequência < (30) (90) > é “suportada” pelos consumidores 1 e 4. O consumidor 4 comprou os itens (40,70) entre os itens 30 e 90, mas ele suporta a

sequência < (30) (90) > pois os padrões buscados não precisam ser necessariamente contínuos. O padrão sequencial < (30) (40,70) > é "suportado" pelos consumidores 2 e 4. O consumidor 2 comprou o item 60 junto dos itens 40 e 70, mas ele também "suporta" este padrão pois (40,70) é um subconjunto de (40,60,70).

Id Cliente	*Timestamp*	Itens Comprados
1	25/jun/93	30
1	30/jun/93	90
2	10/jun/93	10 , 20
2	15/jun/93	30
2	20/jun/93	40 , 60 , 70
3	25/jun/93	30 , 50 , 70
4	25/jun	30
4	30/jun/93	40 , 70
4	25/jul/93	90
5	12/jun/93	90

(a)

Id Cliente	Sequência Cliente
1	< (30) (90) >
2	< (10 20) (30) (40 60 70) >
3	< (30 50 70) >
4	< (30) (40 70) (90) >
5	< (90) >

(b)

Padrão sequêncial com suporte > 25%
< (30) (90) >
< (30) (40 70) >

(c)

Figura 3.2 - (a) Banco de dados de transações ordenado por id cliente e *timestamp*. (b) Banco de dados com as sequências dos clientes. (c) Conjunto de respostas. *Fonte:* (AGRAWAL, 1995)

Diversos algoritmos foram desenvolvidos para tratar este tipo de problema. Alguns deles são: AprioriSOME e AprioriAll (AGRAWAL, 1995), GSP (AGRAWAL, 1996), WAP (PEI, 2000) e PLWAP (EZEIFE, 2005a). A próxima seção descreve algumas características destes algoritmos e do algoritmo selecionado para esta dissertação.

3.4. Algoritmo para Mineração Sequencial

Ao introduzir o problema de minerar padrões sequenciais, Agrawal (1995) também expôs um algoritmo para solucioná-lo. Este algoritmo é chamado AprioriAll e AprioriSOME (uma variação do AprioriAll). Em 1996, Agrawal constatou que o algoritmo desenvolvido por ele se torna inviável à medida que a quantidade de sequências a serem analisadas aumenta e à medida que o tamanho das sequências aumenta.

O algoritmo AprioriAll é dividido em três etapas: encontrar todos os "conjuntos de itens" com suporte mínimo, transformar o banco de dados para que cada transação seja substituída pelo "conjunto de itens" frequentes contidos na transação e gerar os padrões sequenciais. Porém, estas etapas são extremamente custosas computacionalmente (AGRAWAL, 1996).

Devido a esses problemas, Agrawal desenvolveu o GSP, que passou a utilizar a geração de candidatos como uma das etapas para melhorar a performance do algoritmo. Apesar da geração de candidatos executar múltiplas passadas pela base de dados, o tempo de cálculo dos padrões foi reduzido em até 20 vezes. O custo de processamento do GSP é linear em relação à quantidade de sequências e tem uma boa escalabilidade em relação ao tamanho médio das sequências (AGRAWAL, 1996).

Em 2000, Jiawei Han (2000) propôs uma técnica para minerar regras de associação que não utiliza a geração de candidatos. Esta técnica utiliza uma estrutura de dados compacta, chamada de árvore de padrões frequentes ou FP-Tree. Para Jiawei, a geração de candidatos nos algoritmos baseados no Apriori é o grande gargalo desta técnica. Por exemplo, para se obter uma sequência padrão de tamanho 100, $<a_1\ a_2\ a_3\ ...\ a_{100}>$, a quantidade de candidatos gerados é maior que $2^{100} \sim= 10^{30}$ candidatos no total. O algoritmo de Jiawei, FP-Growth, reduz o tempo do GSP em aproximadamente 10 vezes e este tempo diminuía ainda mais quando o tamanho das sequências aumentava. Porém FP-Growth não considera a ordem das transações para montar as sequências (EZEIFE, 2005a).

Ainda no ano 2000, Pei et al. (2000) desenvolveu um algoritmo chamado WAP, que tinha como principal objetivo descobrir padrões de acesso a páginas web. O WAP armazena os dados em uma estrutura chamada WAP-Tree, que é uma árvore de prefixos similar à FP-Tree, mas voltada para mineração sequencial.

Em 2005, Ezeife e Yi Lu aprimoram a WAP-Tree de forma a não ser mais necessária a reconstrução recursiva de WAP-Trees intermediárias durante a mineração da WAP-Tree original em busca de padrões. Este algoritmo se chama PLWAP, Pre-Ordered Linked WAP-Tree (EZEIFE, 2005a).

Tanto o WAP quanto o PLWAP são voltados para o domínio de descoberta de conhecimento em logs de páginas da web, mais precisamente, eles verificam as páginas acessadas por um usuário em um determinado espaço de tempo e geram os padrões sequenciais de acesso. Ezeife e Yi Lu (2005b) implementaram os algoritmos GSP, WAP e PLWAP em código aberto ainda em 2005.

Vale observar que o WAP e o PLWAP funcionam apenas com itens simples. Ou seja, cada usuário só poderá acessar uma página para um mesmo *timestamp*. Mapeando para o contexto de comércio eletrônico, é como se cada transação efetuada por um usuário pudesse conter somente um item.

Ezeife (2005b) executou vários experimentos para os algoritmos implementados, nos quais destacamos três. O primeiro experimento utilizava um banco de dados pequeno, de 40 mil transações (Figura 3.2-3-3.a), o segundo um banco de dados médio, de 200 mil transações (Figura 3.2-3-3.b) e o terceiro um banco de dados grande 1 milhão de transações (Figura 3.2-33). Todos os experimentos utilizaram um suporte que varia entre 0,05% e 15%. A quantidade de padrões encontrada é representada por Fp. Pela Figura 3.2-33.b, para um suporte de 0,5%, é observado que o PLWAP consegue ser até 900 vezes mais rápido que o GSP.

Alg	Tempo de execução (em segundos) para suportes diferente (%)						
	0.05 Fp= 2729	0.1 Fp= 1265	0.5 Fp = 268	1 Fp= 111	5 Fp= 23	10 Fp= 10	15 Fp 0
GSP	6663	3646	1054	636	157	30	1
WAP	149	66	20	8	3	1	1
PLWAP	54	26	7	3	1	1	1

(a)

Alg	Tempo de execução (em segundos) para suportes diferente (%)						
	0.05 Fp= 2630	0.1 Fp= 1271	0.5 Fp = 20	1 Fp= 114	5 Fp= 23	10 Fp= 10	15 Fp 0
GSP	34275	10021	11742	3320	785	150	4
WAP	145	78	27	17	7	5	4
PLWAP	78	47	13	9	5	4	4

(b)

Alg	Tempo de execução (em segundos) para suportes diferente (%)						
	0.05 Fp= 2646	0.1 Fp= 1269	0.5 Fp = 273	1 Fp= 116	5 Fp= 23	10 Fp= 10	15 Fp 0
GSP	73084	41381	12854	7358	1715	328	9
WAP	387	100	41	27	14	10	9
PLWAP	236	87	28	17	10	9	8

(c)

Figura 3.2-3 - Tempo de execução para um conjunto de dados com diferentes suporte ((a) -banco de dados pequeno, (b) – banco de dados médio e (c) – banco de dados grande). *Fonte:* (EZEIFE, 2005b)

Na próxima seção é apresentada a modificação necessária no algoritmo implementado por Ezeife (2005b) para permitir seu uso nesta dissertação.

3.5. Algoritmo Selecionado

Para esta dissertação, foram realizados testes com os três algoritmos implementados por Ezeife (2005b), GSP, WAP e PLWAP, para uma pequena base de dados contendo 15 transações e foi observado, como esperado, que todos exibiram os mesmos resultados, ou seja, geraram os mesmos padrões. Como a base de dados utilizada por esta dissertação tem aproximadamente 23 mil transações e o suporte utilizado é de aproximadamente 0.5%, o algoritmo adotado como base para minerar padrões sequenciais foi o mais rápido, o PLWAP.

O algoritmo PLWAP implementado por Ezeife (2005b) requer um arquivo de entrada e gera um arquivo de saída. Ao iniciar a execução, o algoritmo pede que seja inserido o suporte mínimo a ser considerado, que compreende um valor entre 0 e 1.

O arquivo de saída contém as sequências que obtiveram suporte maior que o determinado no início da execução do programa. Vale observar que o algoritmo retorna todas as sequências, não somente as sequências máximas.

Foi necessário adaptar a implementação em questão para incluir o suporte de cada sequência no arquivo de saída, permitindo que os resultados sejam classificados por ordem de relevância.

3.6. Considerações Finais

Esta seção mostrou a técnica de mineração de dados relacionada à geração de regras de associação e comentou sobre alguns algoritmos que implementam essas técnicas.

Foram testados os algoritmos GSP, WAP e PLWAP e todos produziram os mesmos padrões de sequência, diferenciando-se apenas no tempo de execução. Em todos os casos, os arquivos de saída contêm apenas as sequências, sem informações do suporte delas. Foram feitas algumas alterações no algoritmo para que fosse possível exibir o suporte ao lado das sequências, pois essa informação é necessária para classificar as sequências que serão exibidas ao usuário.

O próximo capítulo irá mostrar como as bases foram geradas, como os workflows foram “transformados” em transações para que fosse possível a utilização destes algoritmos e como são geradas as recomendações.

Capítulo 4 - O Sistema de Recomendação Proposto

4.1. Introdução

Após o levantamento da necessidade de criação de um sistema de recomendação para workflows, foi identificada, também, a necessidade de adaptar uma técnica de mineração de dados para ser utilizada neste sistema.

Neste capítulo, é apresentada a técnica de mineração de dados escolhida, assim como as adaptações desta técnica, do contexto de comércio eletrônico, para o contexto de workflows. Também são apresentados os passos necessários para construir este sistema de recomendação para workflows.

A seção 4.2 apresenta a visão geral da abordagem. A seção 4.3 apresenta as adaptações e considerações necessárias para o entendimento desta abordagem. A seção 4.4 apresenta a fase de preparação dos dados e a seção 4.5 apresenta a fase de consulta e recomendação. A seção 4.6 finaliza o capítulo com as considerações finais.

4.2. Visão Geral da Abordagem

A visão geral da abordagem proposta nesta dissertação está ilustrada na Figura 4-1. Em virtude da demanda de processamento de algumas etapas do processo de recomendação, esta abordagem foi dividida em duas fases: 1) preparação e 2) consulta e recomendação.

A **preparação** é a fase inicial, responsável pelo preprocessamento dos dados provenientes dos workflows previamente desenvolvidos, de forma a permitir a utilização do algoritmo de mineração selecionado. Esta fase também é responsável por aplicar o algoritmo de mineração e extrair as regras desses dados. A fase de preparação é a que demanda mais tempo, pois é nela que são criados os modelos de dados. A fase

de **consulta e recomendação** é responsável por receber as consultas dos usuários e recomendar os trechos de workflows que sejam mais adequadas ao que se procura.

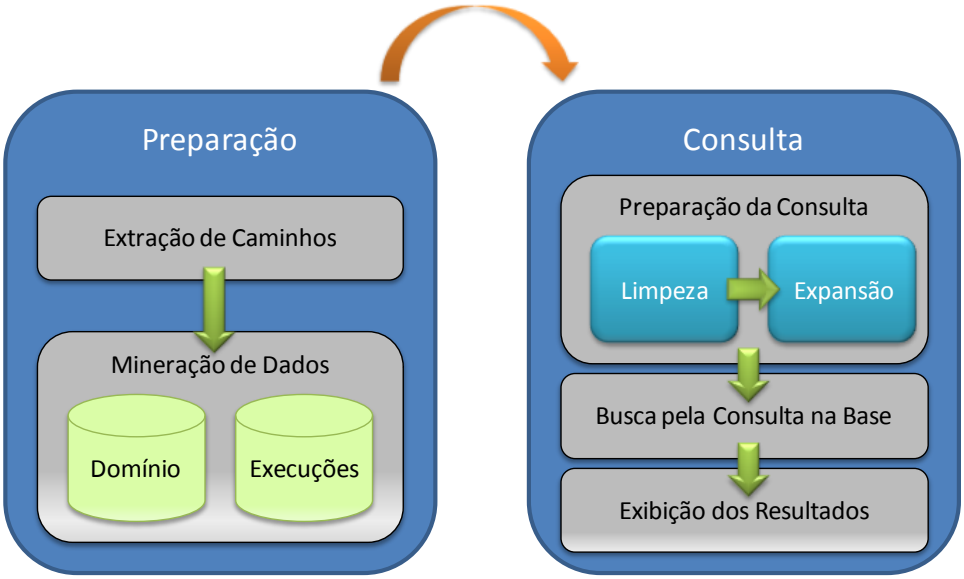


Figura 4-1 - Visão Geral da Abordagem

Os detalhes de cada fase, assim como as adaptações do algoritmo de mineração de dados selecionado, são tratados nas próximas seções.

4.3. Adaptação do Problema de Mineração de Dados

O estudo sobre as técnicas de mineração de dados, realizado no capítulo 3, mostrou que no domínio de comércio eletrônico, a descoberta de regras de associação, para eventos não temporais, e a mineração sequencial, para eventos temporais, são algumas das técnicas mais utilizadas e bem sucedidas. Por este motivo e a semelhança do problema de mineração sequencial com o problema de recomendação para workflow, que será exibida nesta seção, a técnica de mineração sequencial foi escolhida.

Esta seção faz, inicialmente, uma analogia entre o domínio de comércio eletrônico e o domínio de workflows e, posteriormente, apresenta uma adaptação do problema de mineração sequencial introduzido por Agrawal (1995) para o contexto de workflows, de forma a obter recomendações utilizando uma técnica consagrada e bem sucedida.

Retornando ao problema de mineração sequencial introduzido por Agrawal (1995), considere a Tabela 4.1, que mostra mais um possível cliente (com identificador igual a 6). A Tabela 4.1.a exibe a data e os itens comprados pelo cliente "6" e a Tabela 4.1.b mostra a sequência de transações dos clientes. O cliente "6" realizou quatro transações, onde comprou apenas um item em cada uma delas.

Tabela 4.1 - Itens simples comprados por um cliente (a). Sequências dos clientes (b). *Fonte:* (AGRAWAL, 1995)

Id Cliente	*Timestamp*	Itens Comprados
6	25/jun/93	30
6	30/jun/93	40
6	10/jul/93	70
6	15/jul/93	90

(a)

Id Cliente	Sequência Cliente
1	< (30) (90) >
2	< (10 20) (30) (40 60 70) >
3	< (30 50 70) >
4	< (30) (40 70) (90) >
5	< (90) >
6	< (30) (40) (70) (90) >

(b)

Graficamente, as transações do cliente "6" podem ser representadas conforme a Figura 4-2. O elemento mais ao topo representa a primeira transação e o elemento mais abaixo representa a última.

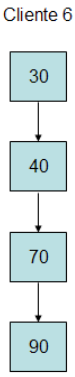
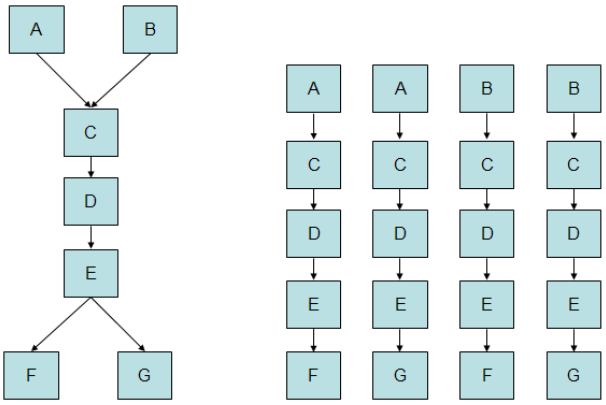


Figura 4-2 - Sequência de compras de um cliente na forma gráfica.

Considere agora a Figura 4-3.a que exibe um workflow contendo sete tarefas. As setas representam o fluxo da execução do workflow, assim como as dependências entre as tarefas. Por essa figura, é observado que a tarefa "A" executa antes da tarefa "C" e que "C" depende diretamente da execução de "A". A tarefa "B" pode ser executada antes, depois ou em paralelo com a tarefa "A".



(a) (b)

Figura 4-3 - Workflow composto por sete tarefas (a) e seus caminhos (b).

Os fluxos ou caminhos presentes no workflow da Figura 4-3.a podem ser divididos conforme a Figura 4-3.b. Cada caminho pode ser visto como uma sequência de tarefas, onde a tarefa mais ao topo é executada primeiro e a mais abaixo por último.

Observando a Figura 4-2 e a Figura 4-3.b, percebe-se que a forma de representar as duas estruturas são iguais. Retornando ao capítulo 2, onde é definido filtragem colaborativa baseada em modelo como sendo a análise de informações históricas para identificar relações entre diferentes itens tal que a compra de um item (ou um conjunto deles) geralmente implica na compra de outro item (ou outro conjunto de itens), também é possível identificar uma semelhança com o problema de recomendação para workflows, que pode ser definido como: Dado uma tarefa T_i (ou sequencia de tarefas), deseja-se saber qual tarefa T_j (ou sequência delas) pode suceder T_i, tomando como referência um histórico de workflows previamente desenvolvidos.

A abordagem desta dissertação propõe um mapeamento dos conceitos de comércio eletrônico para workflows científicos, conforme a Tabela 4.2, para que seja possível montar uma analogia entre os dois domínios.

Tabela 4.2 - Mapeamento dos conceitos de comércio eletrônico para workflow.

Domínio	Conceito			
Comércio Eletrônico	Cliente	Timestamp	Item	Sequencia
Workflow Científico	Usuário	Dependência	Tarefa	Caminho

Com essa analogia entre comércio eletrônico e workflows, foi possível aproveitar todos os conceitos e algoritmos da mineração sequencial e adaptar o problema introduzido por Agrawal (1995) da seguinte maneira: dado um banco de dados *BD* de caminhos de workflows (sequência de tarefas), o problema de minerar padrões sequenciais em workflows é encontrar as sequências máximas (i.e., sequências que não estão contidas em nenhuma outra sequencia) dentre todas as sequências que atingiram o suporte mínimo determinado. Cada uma dessas sequências máximas encontradas será um *caminho padrão*. Para isso, é necessário transformar os workflows em caminhos e aplicar o algoritmo de descoberta de padrões sequenciais para se obter as recomendações (caminhos padrão).

Como dito na seção anterior, devido ao tempo necessário para extrair os caminhos dos workflows e calcular os caminhos padrão, a abordagem foi dividida em duas fases. A primeira fase desta abordagem, preparação, está descrita na próxima seção.

4.4.Fase de Preparação

Esta fase da abordagem é responsável por preparar os dados provenientes de workflows previamente desenvolvidos, de forma que seja possível aplicar um algoritmo de mineração sequencial e obter caminhos padrão de workflows. Além do algoritmo de mineração, outros algoritmos são necessários para a realização desta fase, conforme apresentado na Figura 4-4, que representa a visão geral desta fase. Vale ressaltar que a

fase de preparação será executada poucas vezes, apenas quando houver a necessidade de criar uma nova base de caminhos padrão.

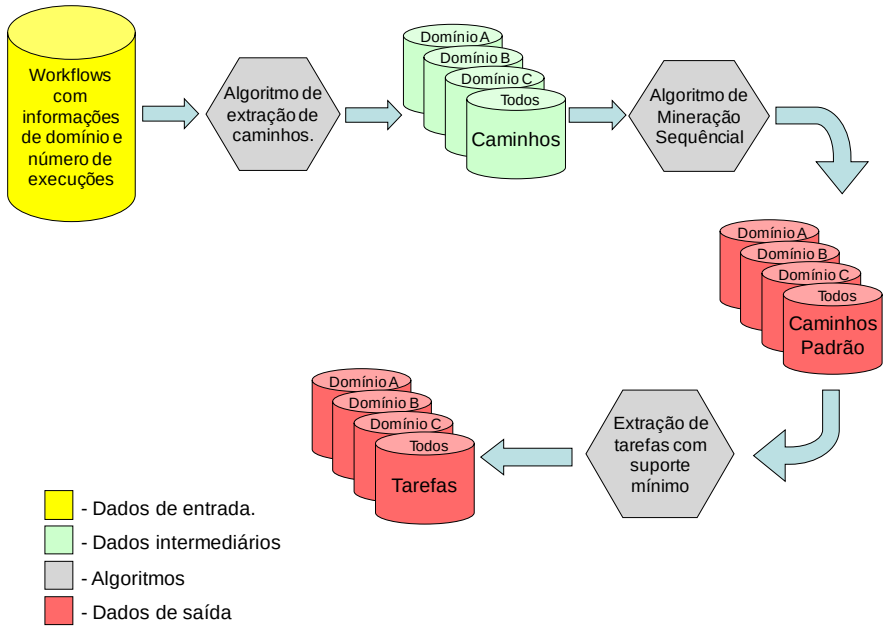


Figura 4-4- Visão Geral da Fase de Preparação

4.4.1. Extração dos Caminhos Máximos

A primeira atividade desta fase é a extração de todos os caminhos presentes nos workflows. Os caminhos encontrados serão armazenados de acordo com o domínio no qual o workflow pertence. Por exemplo, bioinformática, modelagem 3D, entre outros.

Considerando novamente a Figura 4-3.b, os caminhos exibidos são chamados de caminhos completos, ou caminhos máximos. Analogamente a uma sequência máxima, um caminho máximo é o caminho que não está contido em nenhum outro caminho em um mesmo workflow. Esta abordagem, assim como a abordagem proposta por Agrawal (1995), considera somente os caminhos máximos como entrada do algoritmo de mineração, pois evita a contagem repetitiva de caminhos intermediários em um mesmo workflow. Vale ressaltar, que o algoritmo de mineração sequencial detecta as subsequências automaticamente. Para ilustrar esta situação, considere que os caminhos máximos A→B→C e A→B→D aparecem em grande parte dos workflows, mas

individualmente não atingem o suporte mínimo. O próprio algoritmo de mineração detecta que A→B é uma sequência que deve ser recomendada, mesmo não sendo um caminho máximo.

4.4.2. Separação dos Caminhos por Domínio

Conforme ilustrado na Figura 4-4, as bases são separadas de acordo com o domínio no qual o workflow pertence.

A separação de caminhos por domínio permite inferir uma recomendação mais precisa, pois aumenta o suporte dos caminhos pertencentes ao domínio que se deseja a recomendação. Para ilustrar este problema, considere os caminhos da Figura 4-5. Considere que o caminho X, do domínio "bioinformática", ocorre 100 vezes na base, ou seja, 100 workflows utilizaram este caminho em sua composição, e o caminho Y, do domínio "petróleo", ocorre 500 vezes. Considere também que o total de caminhos na base seja 1000, sendo 200 do domínio de "bioinformática" e 800 do domínio de "petróleo".

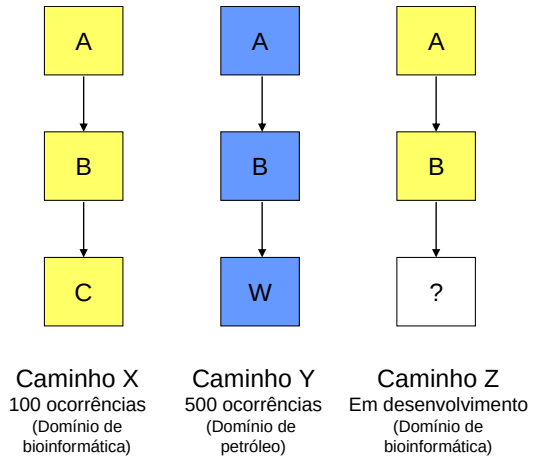


Figura 4-5 - Caminhos separados por domínio.

Considere que um usuário está desenvolvendo um workflow no domínio de bioinformática e ao chegar ao estágio de desenvolvimento representado pelo caminho Z da Figura 4-5, ele solicita uma recomendação ao sistema com um suporte mínimo de

20%. Caso a base considerada seja a base completa, sem separação por domínio, o caminho que o sistema retorna é o caminho Y (suporte de 50%), pois este é o único caminho que contém o caminho Z e tem suporte acima de 20%. O caminho que deveria ser recomendado ao usuário, caminho X, obteve um suporte de apenas 10% e foi excluído da recomendação.

Considerando a separação das bases por domínio e selecionando o domínio da bioinformática, o caminho X passa a ter um suporte de 50% e pode ser exibido ao usuário como um caminho padrão. Isto permite mostrar ao usuário que a próxima tarefa a ser inserida em seu workflow deve ser a tarefa C, não a tarefa W.

4.4.3. Atribuição de Pesos

Outra característica desta abordagem, para que as recomendações sejam mais precisas e de melhor qualidade, é a atribuição de pesos aos workflows. Workflows que são mais utilizados, mais executados, provavelmente têm um maior grau de confiabilidade na disposição de suas tarefas e, consequentemente, contém caminhos mais confiáveis. Estes workflows recebem um peso maior para que seus caminhos sejam recomendados com um valor de suporte mais alto.

Considere a Figura 4-6, onde um usuário deseja obter recomendação para o caminho Z em desenvolvimento. O caminho X foi extraído de apenas um workflow e este workflow foi executado apenas uma vez. O caminho Y também foi extraído de apenas um workflow, porém, este workflow foi executado 100 vezes. Caso não seja considerado o peso, ou o número de execuções, ao se pedir uma recomendação, o sistema retornará o caminho X e caminho Y com o mesmo suporte, porém, é de se esperar que os caminhos de um workflow que tenha sido executado mais vezes sejam mais confiáveis e, consequentemente, tenham um suporte maior.

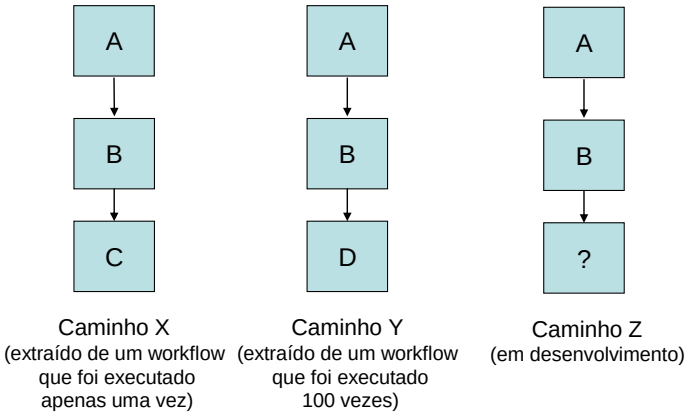


Figura 4-6 - Caminhos separados por execução.

Vale observar que a atribuição de pesos aos workflows pode gerar efeitos colaterais, pois pode alterar drasticamente o suporte de todos os caminhos da base e, muitas das vezes, até excluir caminhos do conjunto de caminhos padrão.

Para o exemplo da figura anterior, considerando que a base contém somente esses dois caminhos, se o suporte selecionado pelo usuário for 10%, ambos os caminhos, X e Y, aparecem na recomendação quando não se considera o peso, porém, ao se adicionar a variável peso no cálculo da recomendação, o suporte do caminho X fica em menos de 1% e este caminho é excluído da recomendação.

Para a atribuição de pesos, durante a atividade de extração de caminhos, é contado o número de vezes que o workflow foi executado completamente, e para cada execução, todos os seus caminhos serão inseridos novamente na base de caminhos.

4.4.4. Aplicação do Algoritmo de Mineração Sequencial

Com os caminhos extraídos dos workflows e inserido nas bases de caminhos, a próxima atividade da preparação é executar o algoritmo de mineração sequencial sobre todas essas bases (base completa e bases dos domínios). O algoritmo de mineração gera um conjunto de caminhos padrão para cada uma dessas bases. Este conjunto de caminhos padrão servirá como base de consulta para gerar as recomendações.

Outra saída da fase de preparação é uma lista de elementos distintos, únicos, chamada de “lista de sequências mínimas”. Esta lista contém todas as tarefas presentes em caminhos que obtiveram um suporte mínimo. Ou seja, para cada tarefa de um caminho padrão, é verificado se ela já está presente nesta lista e, caso não esteja, a tarefa é inserida. Esta lista serve para reduzir o escopo da busca na fase de consulta e recomendação, e será tratada com mais detalhes na próxima seção.

Finalizada a fase de preparação, a próxima etapa é a fase de consulta e recomendação. A consulta utiliza tanto os **dados de saída** (Figura 4-4) gerados pelo algoritmo de mineração sequencial quanto os **dados intermediários** (Figura 4-4) gerados pelos demais algoritmos da fase de preparação.

4.5. Fase de Consulta e Recomendação

A fase de consulta e recomendação é a fase responsável por receber as consultas dos usuários e retornar, ordenadamente, as melhores recomendações. As recomendações nada mais são do que as sequências mais frequentes encontradas pelo algoritmo de mineração sequencial, que na prática, são os caminhos mais frequentes encontrados nos workflows.

Considerando que as bases com os caminhos mais frequentes já estão carregadas e o usuário já escolheu de quais domínios deseja obter recomendações, o usuário já pode iniciar o processo de consulta para receber as recomendações.

A Figura 4-7 ilustra a fase de consulta e recomendação.

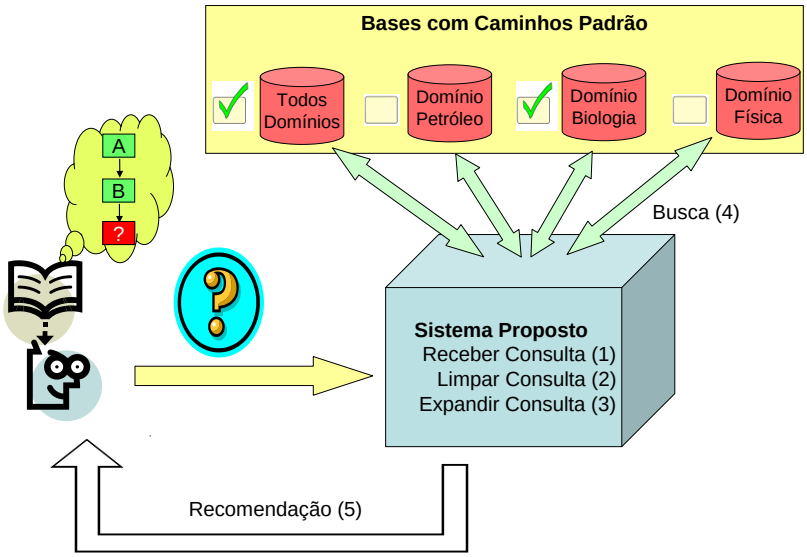


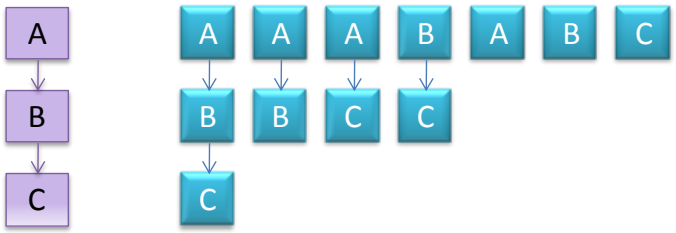
Figura 4-7 - Etapas da Consulta e Recomendação.

A fase de consulta e recomendação se inicia toda vez que um usuário envia uma consulta ao sistema. Esta consulta, tratada pela etapa Receber Consulta (1), é um caminho, mais precisamente, um trecho de workflow, que pode conter uma ou mais tarefas. No exemplo da figura, o trecho de workflow enviado como consulta é "A→B".

O próximo passo, Limpar Consulta (2), é responsável por eliminar desta consulta as tarefas que não estão presente na lista de sequências mínimas, ou seja, eliminar as tarefas que não aparecem em nenhum caminho padrão. Caso alguma tarefa da consulta seja descartada, a consulta não é prejudicada, visto que a busca por caminhos que contêm essa tarefa retorna vazio. A redução desta sequência de entrada também diminui os recursos computacionais gastos nas próximas etapas.

Para ilustrar a etapa Limpar Consulta (2), considere que o usuário deseja obter recomendação para o trecho de workflow A→B→Y→C, onde Y é uma tarefa que não está presente em nenhum caminho padrão. Após a etapa Limpar Consulta (2), a consulta passa a ser A→B→C, visto que Y não está presente na lista de sequências mínimas. Esta limpeza permite que sequências do tipo A→B→C→(X), onde (X) é um caminho contendo uma ou mais tarefas, seja recomendada ao usuário, o que não seria possível caso a tarefa Y estivesse presente.

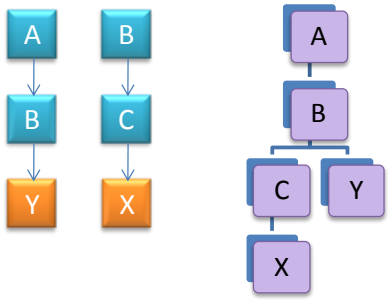
A terceira etapa da fase de consulta e recomendação, Expandir Consulta (3), consiste em encontrar as possíveis subsequências contidas na sequência de consulta resultante da etapa Limpar Consulta (2). Para melhor ilustrar, considere a Figura 4-8, onde a Figura 4-8.a apresenta uma possível consulta e a Figura 4-8.b apresenta todas as subsequências geradas a partir da consulta de entrada. Com este procedimento, é possível gerar recomendações para todos os ramos de um workflow.



(a) (b)

Figura 4-8 - Consultas (a) e sub-consultas (b).

Para ilustrar este tipo de recomendação, considere a consulta do exemplo anterior. Na Figura 4-9.a são exibidas as recomendações encontradas pelo sistema para o subcaminho A→B, que é a tarefa Y, e para o subcaminho B→C, que é a tarefa X. O workflow do usuário, após a aceitação das recomendações, ficaria como apresentado na Figura 4-9.b.



(a) (b)

Figura 4-9 - Recomendações (a) e workflow final (b).

A etapa Expandir Consulta (3) também favorece tarefas que tem um alto grau de sinergia, mesmo contendo tarefas entre elas. Para ilustrar, considere que se deseja obter recomendação para o caminho A→W→B, onde todas as três tarefas estão presentes na

lista de sequências mínimas, porém nenhum caminho máximo contém essas três tarefas ao mesmo tempo. Considere também que o caminho A→B→C tem um suporte muito alto. Caso a consulta seja A→W→B, o sistema não retornaria nenhum resultado, porém, com a execução da etapa (3), a consulta seria dividida em A→W, W→B e A→B, e se obteria uma recomendação para A→B, a tarefa C, resultando no caminho A→W→B→C.

A próxima etapa desta fase, Busca (4), é responsável por pesquisar nas bases selecionadas quais caminhos padrão contém alguma das combinações de caminhos geradas no passo anterior.

Um caminho P_1 está contido em um caminho P_2, se todas as tarefas de P_1 estão contidas em P_2 e estão na mesma ordem, sem necessariamente estarem conectadas diretamente. Por exemplo, C → D está contido em A → B → C → D, assim como A → C → D também está. Porém, D → A → B não está contido em A → B → C → D. Desta forma, é possível obter recomendação para as duas extremidades do caminho, ou seja, nos dois sentidos do workflow. O produto final desta etapa é uma lista de caminhos padrão ordenados por suporte e separados por domínio.

A última etapa da fase de consulta e recomendação, Recomendação (5), consiste em exibir, de forma ordenada, os resultados obtidos na etapa anterior ao usuário.

4.6. Considerações finais

Este capítulo apresentou a visão geral da abordagem proposta. Também detalhou todas as fases envolvidas no processo de recomendação, assim como a técnica de mineração de dados selecionada e as adaptações necessárias desta técnica.

O próximo capítulo trata da implementação desta abordagem, assim como dos resultados obtidos na recomendação com a utilização deste sistema de recomendação.

Capítulo 5 - Implementação e Avaliação

5.1. Introdução

Este capítulo apresenta a implementação de um protótipo e a avaliação da abordagem proposta. Em conjunto com a avaliação deste protótipo, é realizada a avaliação de outra abordagem, o VisComplete, de forma a obter um comparativo entre os dois sistemas. O foco principal desta avaliação é em aspectos relacionados à precisão dos resultados obtidos e ao desempenho dos algoritmos.

Inicialmente, na seção 5.2, é feito um detalhamento técnico da Fase de Preparação, descrevendo como foi implementado cada elemento da Figura 4-4. Na seção 5.3, é detalhada a Fase de Consulta e Recomendação. Posteriormente, na seção 5.4, é descrita a avaliação da abordagem proposta e da ferramenta VisComplete. Por último, na seção 5.5, são feitas as considerações finais deste capítulo.

5.2. Detalhamento Técnico da Fase de Preparação

A implementação, feita na linguagem Python, utiliza um banco de dados relacional para o armazenamento de workflows previamente desenvolvidos e arquivos texto para armazenar os dados resultantes da Fase de Preparação.

Os workflows desenvolvidos são armazenados na forma de conexões, porém, como dito no Capítulo 4, os workflows precisam ser separados em caminhos para que seja possível a utilização do algoritmo de mineração sequencial. Cada conexão de um workflow contém duas tarefas, uma origem e uma destino. Para ilustrar, considerando o caminho A→B→C, as suas conexões são A→B e B→C, onde o antecedente da relação é a origem e o consequente o destino.

A Figura 5-1 apresenta um diagrama entidade-relacionamento (DER) da estrutura que representa os workflows. Cada entrada da entidade “workflow” contém

uma ou mais conexões e cada conexão é composta por duas tarefas. Um conjunto de conexões forma, implicitamente, um caminho. A entidade “tarefa” contém todas as tarefas que podem ser utilizadas nos workflows e a entidade “domínio” contém todos os domínios no qual os workflows pertencem.

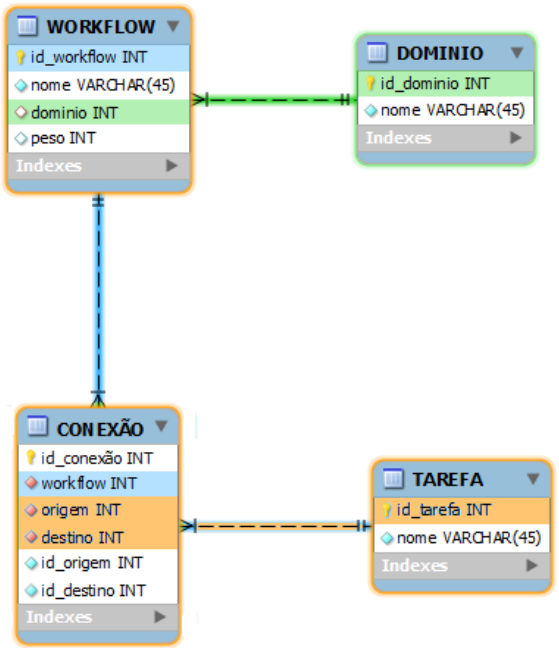


Figura 5-1 - Diagrama entidade-relacionamento da abordagem.

A entidade “workflow” contém os atributos nome, peso e domínio, que representam, respectivamente, o nome do workflow cadastrado, o número de vezes que o workflow foi executado completamente e o domínio do workflow. A entidade “domínio” e “tarefa” contêm o atributo nome, que é o nome do domínio e nome da tarefa, respectivamente, além dos identificadores únicos da tarefa e de domínio. As entidades “conexão” contém os atributos workflow, que representa o “dono” da conexão; origem e destino, que são as tarefas origem e destino, respectivamente; e id_origem e id_destino, que será explica a seguir.

Para explicar a necessidade dos atributos id_origem e id_destino, considere o workflow da Figura 5-2, onde o valor entre parênteses corresponde ao identificador único da tarefa (id_tarefa).

Os caminhos deste workflow são: A→C→D→A e B→C→D→A. Considerando apenas o primeiro caminho, as conexões são A→C, C→D e D→A. Caso não haja um identificador para a tarefa antecedente "A" e a tarefa consequente "A" que identifique cada instância, este caminho pode ser interpretado, de forma errônea, como um loop.

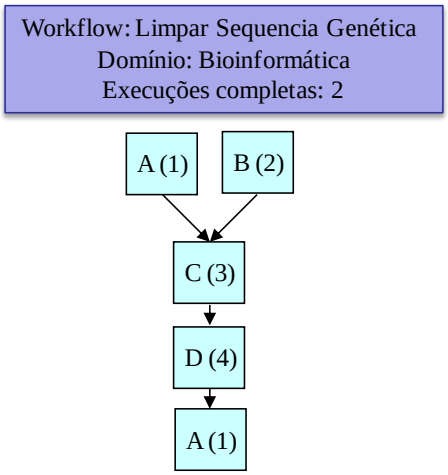


Figura 5-2 - Workflow com identificador nas tarefas.

A Figura 5-3 faz o mapeamento do workflow presente na Figura 5-2 para o diagrama proposto na Figura 5-1.

CONEXÃO

workflow	origem	destino	id_origem	id_destino
1	1	3	1	2
1	3	4	2	3
1	4	1	3	4
1	**2**	**3**	**5**	**2**
1	**3**	**4**	**2**	**3**
1	**4**	**1**	**3**	**4**

TAREFA

id_tarefa	nome
1	A
2	B
3	C
4	D

WORKFLOW

id_workflow	nome	dominio	peso
1	Limpar Sequencia Genética	1	2

DOMINIO

id_dominio	nome
1	Bioinformática

Figura 5-3 - Tabelas preenchidas com workflow exemplo.

Como dito anteriormente, o workflow da Figura 5-2 contém dois caminhos, A → C → D → A, cujas conexões estão representadas pelas linhas em branco na tabela CONEXÃO da Figura 5-3, e B → C → D → A, representado pela cor cinza. É possível observar que, mesmo a tarefa A tendo o mesmo identificador 1, ao se montar a tabela

com as conexões, o identificador origem/destino é diferente, garantindo que as duas tarefas são instâncias distintas.

Como os workflows são armazenados como conexões, foi necessário criar um algoritmo para converter essas conexões em caminhos máximos. Esta é a primeira atividade da Fase de Preparação exibida na Figura 4-4.

A primeira atividade do algoritmo para encontrar caminhos é encontrar as tarefas que são folhas, ou seja, as tarefas que, para um mesmo workflow, são destino, mas não são origem de nenhuma conexão. O resultado desta atividade é uma lista contendo todas as tarefas folha. A segunda atividade é encontrar a tarefa pai de cada folha e para cada tarefa pai é encontrada a sua tarefa pai, e assim sucessivamente, até encontrar a primeira tarefa do caminho. O resultado desta atividade é a lista de caminhos de todos os workflows da base.

Os caminhos extraídos dos workflows são armazenados em um arquivo texto que serve de entrada para o programa de mineração sequencial. O algoritmo PLWAP, implementado por Ezeife (2005b), necessita de um arquivo de entrada e gera um arquivo de saída.

O arquivo de entrada para o algoritmo é composto por uma estrutura padrão, onde cada linha contém a identificação única do caminho, o tamanho da sequência e a sequência propriamente dita. A sequência é composta por números inteiros que correspondem aos identificadores das tarefas. Um exemplo de uma linha do arquivo é:

34 6 8 79 12 13 14 1

Onde o identificador do caminho é 34, o número 6 corresponde ao tamanho da sequencia e os outros números compõem a sequência de tarefas. Um trecho do arquivo de entrada para o programa é apresentado no Apêndice A.

Com todas as entradas do programa de mineração devidamente preenchidas, o programa é executado uma vez para cada base de domínio e, como resultado, gera um arquivo contendo todos os caminhos padrão e seus respectivos suportes para cada base. O arquivo gerado pelo algoritmo PLWAP contém um caminho padrão com seu respectivo suporte por linha. Um exemplo deste arquivo é apresentado no Apêndice B.

A última etapa da fase de preparação é a construção da lista de sequências mínimas de cada domínio, que irá conter as tarefas que estão presentes em pelo menos um caminho padrão gerado pelo programa de mineração. Este algoritmo percorre todas as tarefas de cada caminho padrão e as coloca em uma lista de elementos únicos, esta lista é passada para um arquivo texto que serve de entrada para a próxima fase.

5.3.Detalhamento Técnico da Fase de Consulta e Recomendação

Conforme dito anteriormente, a primeira etapa da Fase de Consulta e Recomendação é receber o arquivo de entrada contendo o caminho ou tarefa para o qual se deseja obter a recomendação.

Após extrair o caminho do arquivo de entrada, este caminho passa para a etapa de Limpeza. Esta etapa tem como objetivo eliminar da consulta as tarefas que não estão presentes em um caminho padrão. O algoritmo de limpeza de consulta é um algoritmo que percorre a lista de tarefas para a qual se deseja realizar uma recomendação e para cada tarefa desta lista, é verificado se a mesma está presente também na lista de sequências mínimas. Caso negativo, a tarefa é removida da lista de tarefas consultada.

A próxima etapa da fase de Consulta e Recomendação consiste em expandir a consulta resultante da etapa anterior. Essa expansão, como dito na Seção 4.5, permite a recomendação para todos os ramos do workflow, assim como em todas as direções.

A entrada do algoritmo de encontrar combinação é a consulta do usuário limpa resultante da atividade anterior. O algoritmo de expandir consulta encontra, de forma

recursiva, todas as combinações de uma sequência, sempre respeitando a ordem das mesmas. Para uma consulta limpa A→B→C, o algoritmo retorna uma lista com as seguintes subsequências: [[C], [B], [B→C], [A], [A→C], [A→B], [A→B→C]].

Após a expansão da consulta, as subsequências encontradas servem de parâmetro para realizar as buscas nas bases de caminhos padrão. Esta busca visa encontrar nas bases dos domínios quais sequências padrão contêm as subsequências encontradas na etapa expandir consulta.

A implementação desta abordagem permite selecionar três configurações para o início das buscas. A primeira inicia o processo de busca com a subsequência que contém o maior número de tarefas, que é a própria consulta limpa. Caso não seja encontrada nenhuma recomendação para esta subsequência, selecionam-se as subsequências com o número de tarefas imediatamente menor, e assim por diante. No exemplo do parágrafo anterior, a busca seria iniciada por [A→B→C], posteriormente por [B→C], [A→C] e [A→B], e por último para cada tarefa individualmente, [A], [B] e [C]. A segunda configuração permite executar o processo de busca para todas as subsequências até um limite mínimo de tarefas por subsequência, por exemplo, se o limite fosse duas tarefas, a busca seria realizada para [A→B→C], [B→C], [A→C] e [A→B]. A terceira configuração executa a busca para todas as subsequências encontradas na etapa de expandir busca.

Após a realização das buscas, os caminhos padrão que contém alguma das subsequências geradas pela etapa de expandir consulta são adicionados à lista de recomendações. Esta lista é ordenada pelo suporte e exibida ao usuário.

5.4. Avaliação da Abordagem Proposta

Esta seção trata da avaliação do protótipo implementado neste trabalho, assim como da ferramenta VisComplete. Como o protótipo implementado nesta abordagem

tem por objetivo avaliar o desempenho e a precisão das recomendações, não foi desenvolvida uma interface gráfica, prejudicando um item importante da avaliação do protótipo que é a usabilidade. Portanto, não há figuras do seu funcionamento, porém alguns resultados, assim como os arquivos intermediários, podem ser encontrados nos apêndices.

A avaliação foi dividida em dois grupos: avaliação de requisitos funcionais e avaliação de requisitos não funcionais. Os requisitos funcionais são aqueles que descrevem o comportamento do sistema, suas ações para cada entrada, ou seja, é aquilo que descreve o que tem que ser feito pelo sistema. Os requisitos não funcionais são aqueles que expressam como deve ser feito. Em geral, se relacionam com atributos de qualidade como confiabilidade, desempenho, robustez, etc.

Para esta avaliação, foi utilizada a base de workflows cedida pelos criadores do VisComplete, que é composta por 3343 workflows do domínio de visualização. Estes workflows foram desenvolvidos por 30 estudantes durante um curso de visualização científica na Universidade de Utah, nos Estados Unidos. Também foram utilizados os exemplos distribuídos junto com o programa VisTrails, que também são do domínio de visualização. Como todos os workflows são do domínio de visualização, não há a divisão das bases por domínio durante a fase de preparação, e como também não há informação sobre o número de vezes que cada workflow foi executado, é considerado que todos os workflows têm o mesmo peso.

A avaliação dos requisitos foi realizada em uma máquina com a seguinte configuração: Intel Core 2 Quad 2.6 GHz, 3GB de memória RAM, Disco Rígido de 750GB e sistema operacional Microsoft Windows 7 Professional x64. O banco de dados utilizado foi o MySQL 5.1 Community Server e a linguagem de programação para

implementação do protótipo foi o Python 2.6. Para analisar os gráficos resultantes da avaliação, foi utilizado o software de ajustes de curvas Lab Fit (SILVA, 2004).

5.4.1. Avaliação da Fase de Preparação

Os requisitos avaliados na fase de preparação são todos não funcionais. Entre os requisitos que estão nesta avaliação, encontram-se o tempo gasto para executar todas as etapas da fase de preparação e a quantidade de memória RAM máxima utilizada durante todo o processo de preparação.

Com a dificuldade de se obter bases maiores para realizar a avaliação dos requisitos não funcionais (desempenho), foi necessário replicar a base obtida com os criadores do VisComplete, gerando diferentes bases com quantidades maiores e menores de workflows. Essa multiplicação acabou gerando algumas consequências que favoreceram a nossa abordagem, o que pode comprometer alguns dos testes realizados.

A Tabela 5.1 exibe um sumário da quantidade de workflows, conexões e caminhos por base avaliada, lembrando que a base oficial é a Base 3.

Tabela 5.1 - Quantidades de workflows, conexões e caminhos por base.

	Workflows	Conexões	Caminhos
Base 1	835	6461	2076
Base 2	1671	20321	8294
Base 3	3343	51658	23623
Base 4	6686	103316	47246
Base 5	13372	206632	94492
Base 6	26744	413264	188984
Base 7	53488	826528	377968
Base 8	80232	1239792	755936

A Figura 5-4 exibe um snapshot da fase de preparação para a Base 3, a base original do VisComplete. Inicialmente, os 3343 workflows cadastrados totalizam 51.658 conexões entre tarefas. Ao se extrair os caminhos desses workflows, obtém-se um total de 23.623 caminhos. A próxima etapa, aplicação do algoritmo de mineração sequencial, reduz os caminhos para 2838 caminhos padrão, para um suporte de 0,5%.

Por último, há a extração das 65 tarefas que estão presentes em algum caminho padrão para formar a lista de sequências mínimas. Esta lista e a base de caminhos padrão serão utilizadas para avaliar os requisitos funcionais da fase de consulta e recomendação.

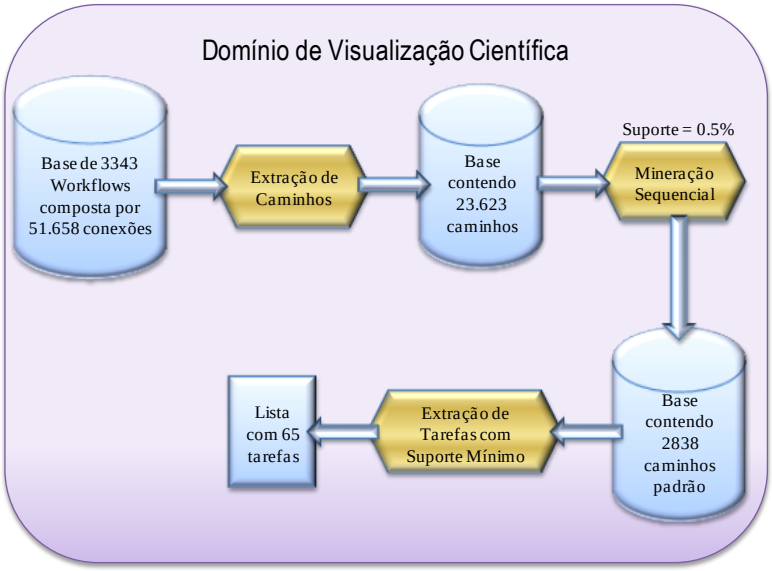


Figura 5-4 - Snapshot da fase de preparação para a base utilizada.

A Tabela 5.2 exibe o tempo gasto e a memória utilizada para realizar a fase de preparação para as diferentes bases, tanto para o VisComplete quanto para o protótipo desta abordagem. O suporte escolhido para a execução do algoritmo PLWAP foi de 2%. A Figura 5-5 apresenta os gráficos dos resultados apresentados na Tabela 5.2.

Tabela 5.2 - Tempo e Memória gastos na fase de preparação.

	VisComplete		Abodagem Proposta	
Bases	Tempo (s)	Memória (MB)	Tempo (s)	Memória (MB)
Base 1	0,47	10	10	9
Base 2	1,5	51	35	15
Base 3	4,4	112	80	30
Base 4	9,68	204	151	54
Base 5	25,3	401	301	100
Base 6	74,87	778	586	193
Base 7	250	1530	1245	378
Base 8	803	2388	1805	540

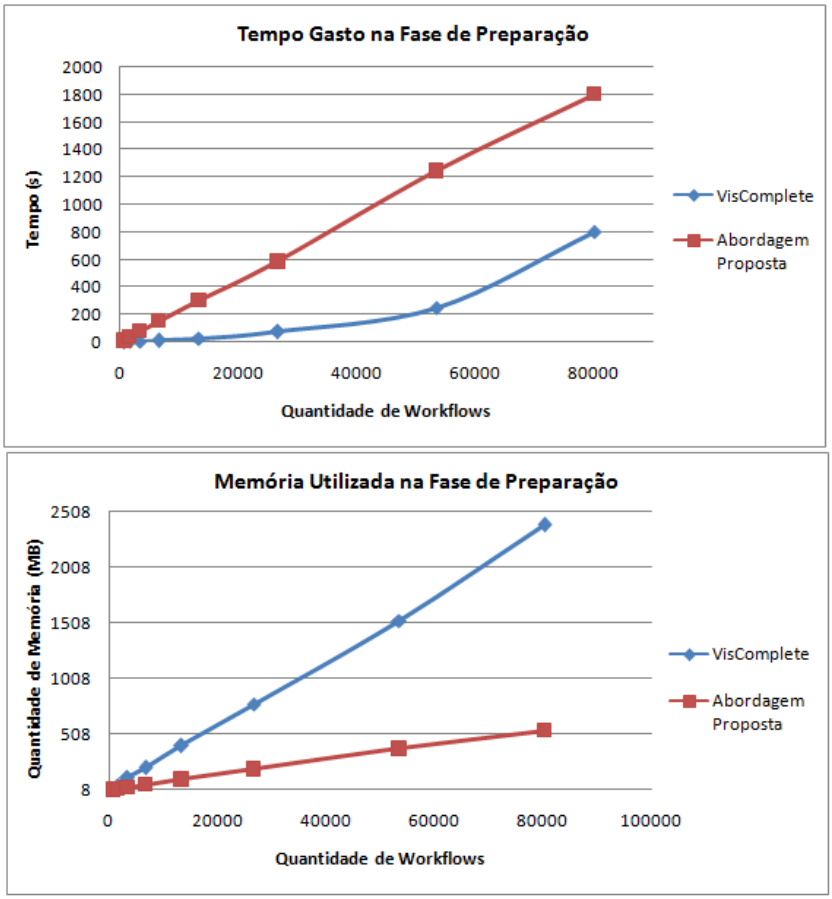


Figura 5-5- Gráficos de utilização de tempo e memória da fase de preparação.

Analisando o gráfico de tempo, é observado que, apesar dos valores nominais em segundos necessários para executar a fase de preparação serem maiores na abordagem proposta, o crescimento é linear, enquanto no VisComplete o crescimento foi polinomial de segundo grau, segundo a análise do software Lab Fit.

Analisando o gráfico de memória utilizada, é observado que ambas as abordagem têm um crescimento linear em relação à quantidade de workflows, porém, como o coeficiente angular da reta de memória utilizada do VisComplete é muito grande, rapidamente foi alcançado o limite de memória RAM da máquina, tornando inviável a utilização do VisComplete para mais de 100.000 workflows.

A análise dos gráficos mostrou que a abordagem proposta leva vantagem em ambos os requisitos avaliados, principalmente em relação ao tempo necessário para

realizar a Fase de Preparação para grandes quantidades de workflows, visto que o tempo varia linearmente com relação à quantidade de workflows.

Como discutido anteriormente, é importante considerar que a fase de preparação é executada poucas vezes e em intervalos de tempo longos, além de poder ser executada de madrugada. Somente a fase de consulta precisa ser executada a cada demanda do usuário, de forma online.

5.4.2. Avaliação da Fase de Consulta e Recomendação

Os requisitos não funcionais avaliados na fase de busca e recomendação são o tempo necessário para carregar as bases e listas geradas na fase de preparação e a memória utilizada para este carregamento. As quantidades de workflows e conexões das bases testadas nesta fase são as mesmas da Tabela 5.1.

A Tabela 5.3 exibe os resultados dos requisitos avaliados e os gráficos destes resultados podem ser observados na Figura 5-6.

Tabela 5.3- Tempo e Memória gastos para carregar as bases.

	VisComplete		Abodagem Proposta	
Bases	Tempo (s)	Memória (MB)	Tempo (s)	Memória (MB)
Base 1	0,23	117	0,09	6,6
Base 2	0,72	126	0,17	7,2
Base 3	1,85	148	0,43	9
Base 4	3,43	184	0,82	9
Base 5	7,5	255	1,61	9
Base 6	18,18	397	3,2	9
Base 7	50,37	681	6,34	9
Base 8	94	963	9,5	9

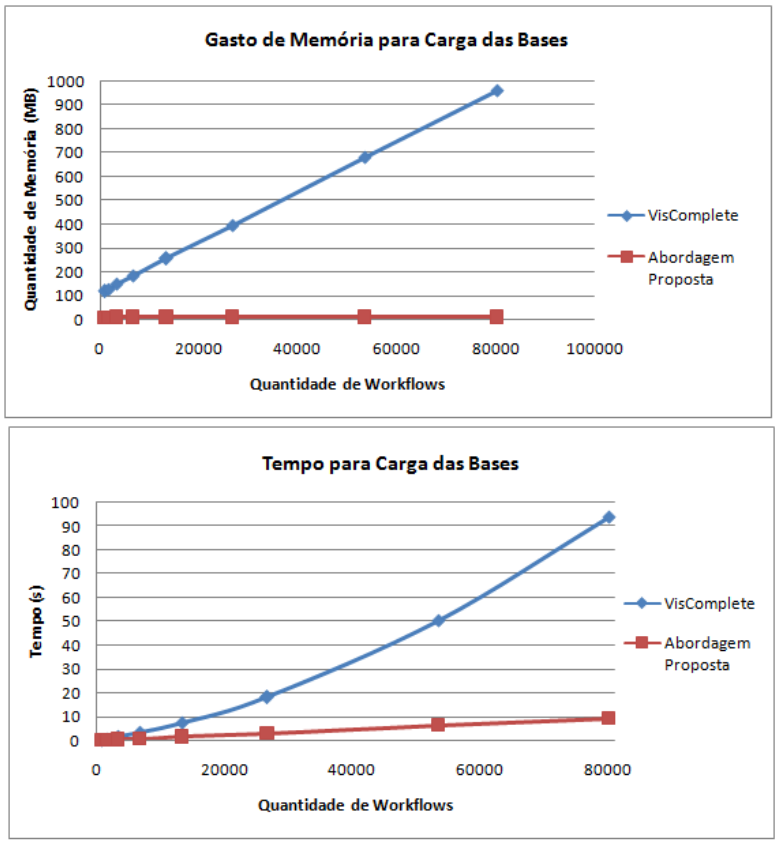


Figura 5-6 - Gráficos de utilização de tempo e memória para carregar as bases preparadas.

Analisando os gráficos, é observado que o gasto de memória no VisComplete tem um comportamento linear em relação ao número de workflows, enquanto o gasto de memória da abordagem proposta é constante a partir da Base 3. Isso ocorre porque como o suporte dos caminhos não muda entre as Bases 3, 4, 5, 6, 7 e 8, o algoritmo de mineração sempre gera os mesmos caminhos padrão e, consequentemente, a quantidade de memória para carregar esses caminhos padrão também não muda.

Em relação ao tempo gasto para carregar as bases, o software Lab Fit mostrou que, para a abordagem proposta, ele tem um crescimento linear com um coeficiente angular bem pequeno, enquanto para o VisComplete, ele obteve um crescimento polinomial de segundo grau.

Esses resultados mostraram que a abordagem proposta leva uma grande vantagem em relação ao VisComplete para realizar o carregamento das bases. Isto

ocorre principalmente por causa da diminuição dos caminhos após a passagem pelo algoritmo PLWAP.

A Tabela 5.4 exibe a quantidade de caminhos total das bases e a quantidade de caminhos padrão gerados pelo PLWAP para um suporte de 2%. Pela tabela, é possível observar que em todos os casos a redução da quantidade de caminhos a ser analisada é maior que 90%. Como dito anteriormente, como as bases 4, 5, 6, 7 e 8 são geradas a partir de replicações da base 3, o suporte dos caminhos não muda e, consequentemente, os caminhos padrão gerados são os mesmos.

Tabela 5.4 - Comparativo entre caminhos e caminhos padrão gerados pelo PLWAP.

	Caminhos	Caminhos Padrão	Redução
Base 1	2076	169	91,86%
Base 2	8294	282	96,60%
Base 3	23623	1039	95,60%
Base 4	47246	1039	97,80%
Base 5	94492	1039	98,90%
Base 6	188984	1039	99,45%
Base 7	377968	1039	99,73%
Base 8	755936	1039	99,86%

As próximas avaliações tratam dos resultados das recomendações, ou seja, dos requisitos funcionais das abordagens. Para a avaliação da precisão das recomendações para ambas as abordagens, a base utilizada foi a base original, não prejudicando a validade desta avaliação.

Para permitir a avaliação dos requisitos funcionais, foi necessário definir uma variável que indicasse o grau de igualdade entre dois caminhos. Esta variável, chamada semelhança *S*, serve de base para escolher qual das recomendações é a melhor, e é explicada a seguir.

Considere que se deseja obter o caminho da Figura 5-7.b, que está sendo chamado de Caminho Objetivo. Ao se buscar por recomendações para a primeira tarefa do caminho objetivo, vtkSphereSource, o sistema retornou dois caminhos, Caminho A,

Figura 5-7.a, e Caminho C, Figura 5-7.c. Considere que o suporte do Caminho C é maior, ou seja, ele é apresentado primeiro ao usuário. O valor do suporte de um caminho não indica se ele é melhor ou pior, mas sim se ele é mais comum ou menos comum. Para verificar qual das recomendações é a mais indicada, é calculada a semelhança entre os caminhos sugeridos e o caminho objetivo, e o que obtiver o maior grau de semelhança é o escolhido. A semelhança *S(Tb)* é o número de tarefas que estão exatamente na mesma sequência que as tarefas do caminho objetivo a partir da tarefa buscada *Tb*. Para a Figura 5-7, a semelhança do Caminho A é igual a dois e a semelhança do Caminho C é igual a um.

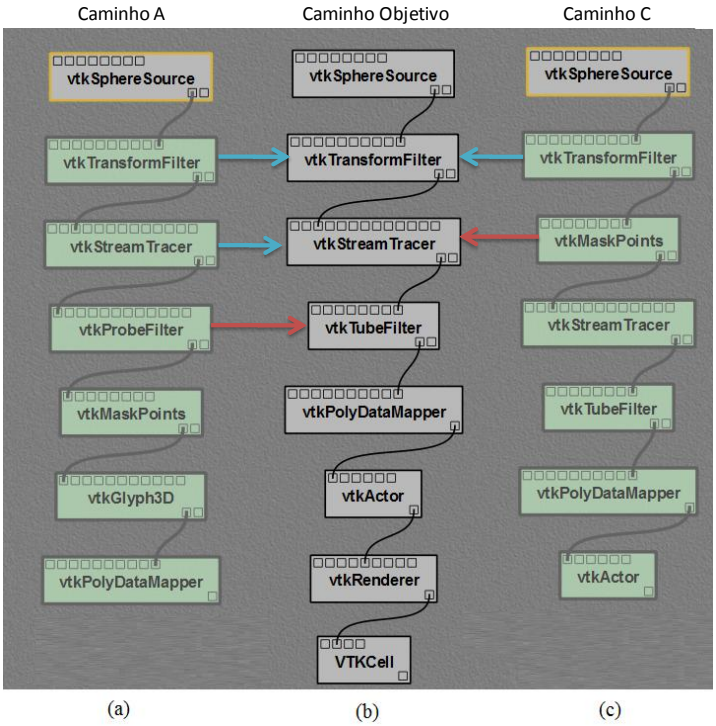


Figura 5-7 - Calculando a semelhança entre caminhos recomendados.

Para verificar qual dos caminhos é a melhor opção de recomendação, é verificada a semelhança do caminho objetivo com os vinte primeiros caminhos recomendados. Para ilustrar como é realizada a avaliação completa das recomendações para um workflow, considere que se deseja construir o workflow apresentado na Figura 5-8.d.

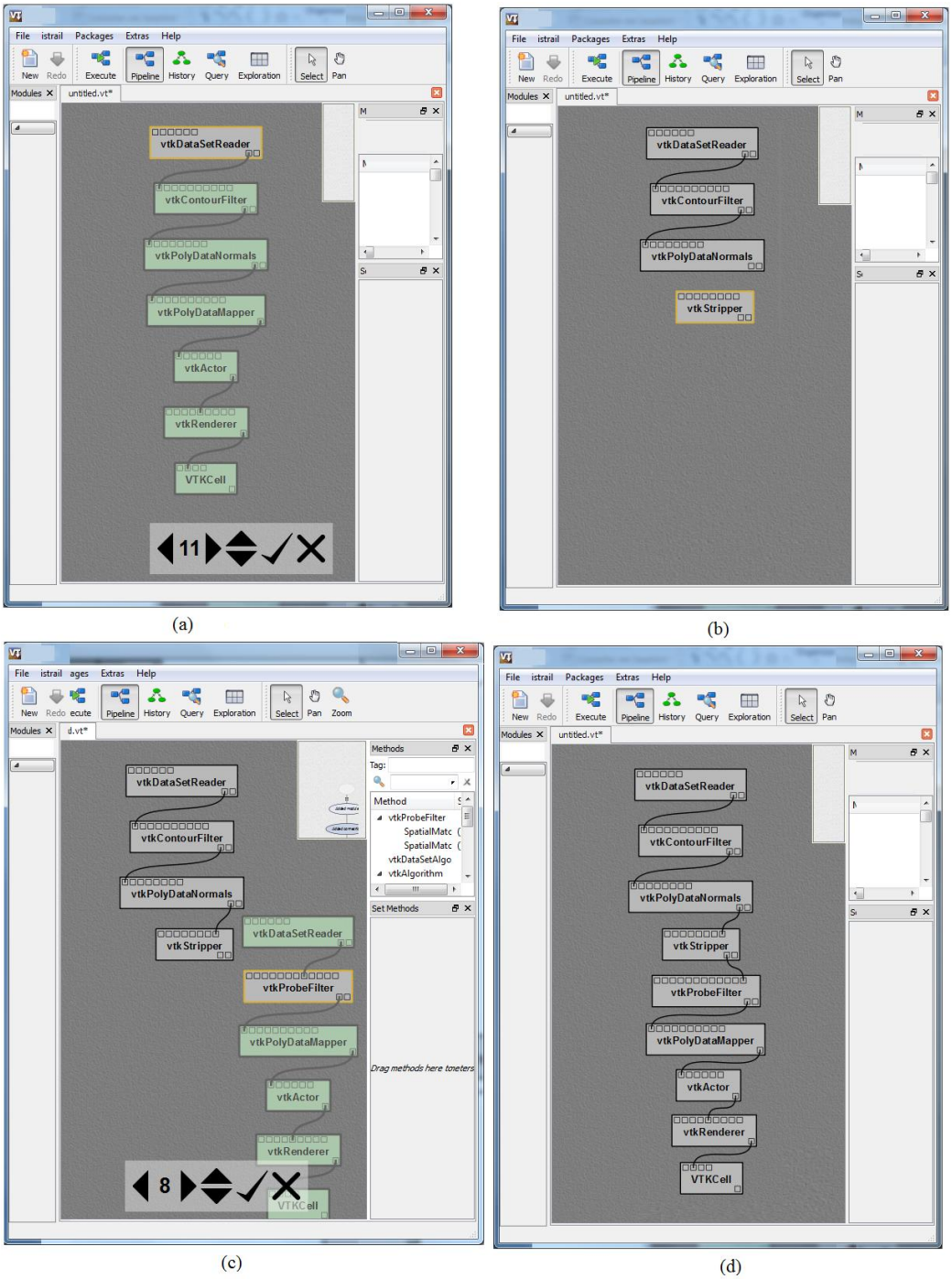


Figura 5-8 - Construção de um workflow com recomendação no VisComplete.

Ao se adicionar a primeira tarefa, vtkDataSetReader, o sistema gera inúmeras recomendações. Ao se navegar pelas recomendações, é detectado que a recomendação que mais se assemelha ao workflow da Figura 5-8.d está na décima primeira posição do ranking, conforme mostra a Figura 5-8.a. O valor da semelhança desta primeira recomendação é igual a dois. O valor obtido na semelhança também serve como ponto

de corte para o caminho recomendado. Após aceitar a recomendação e excluir as tarefas que foram recomendadas além do ponto de corte, é adicionada a segunda tarefa ao sistema, vtkStripper, que, conforme ilustrado na Figura 5-8b, não obtém nenhuma recomendação. Ao se adicionar a tarefa vtkProbeFilter ao workflow em desenvolvimento, Figura 5-8.c, novas recomendações são geradas, e o caminho mais semelhante, com valor de semelhança quatro, está na posição oito da lista. Ao se eliminar a tarefa excedente, vtkDataSetReader, e conectar vtkProbeFilter à vtkStripper o workflow está pronto.

Os requisitos avaliados neste processo de recomendação são o percentual de tarefas que foram recomendadas por workflow, P(R), e a média dos rankings por workflow, M(r).

O percentual de tarefas recomendadas é a soma das semelhanças dos caminhos recomendados para cada tarefa buscada dividida pelo total de tarefas do workflow, conforme mostrado na fórmula a seguir:

$$P(R) = \frac{S(Tb_1) + S(Tb_2) + \ldots + S(Tb_n)}{m}$$

$S(Tb_i)$ é a semelhança do caminho para a i-ésima tarefa buscada de um total de "n" tarefas e "m" é a quantidade total de tarefas no workflow. Para o exemplo da Figura 5-8, P(R)=(2+4)/9=66%. Este percentual indica quanto de trabalho foi economizado no desenvolvimento do workflow, e quanto maior a média dos percentuais para todos os workflows desenvolvidos, mais rápido é o desenvolvimento de novos workflows com o sistema de recomendação utilizado.

A média dos rankings, M(r), avalia a precisão das recomendações para o workflow desenvolvido, e a média de todos M(r) indica o quão preciso é o sistema de recomendação. Na prática, este valor indica em qual posição do ranking é mais provável

de se encontrar a recomendação ideal. O melhor resultado para M(r) é o menor possível e é 1.

Inicialmente, o cálculo de M(r) era somente a média aritmética dos rankings dos caminhos semelhantes, porém, com este cálculo, foi observado que caminhos semelhantes com S(Tb) grandes recebiam o mesmo peso que caminhos semelhantes de apenas uma tarefa (S(Tb)=1). Para contornar este problema, o cálculo de M(r) é representado pela fórmula a seguir:

$$M(r) = \frac{r(Tb_1) + (S(Tb_1) - 1) + r(Tb_2) + (S(Tb_2) - 1) + \ldots + r(Tb_n) + (S(Tb_n) - 1)}{S(Tb_1) + S(Tb_2) + \ldots + S(Tb_n)}$$

Onde $r(Tb_i)$ é o ranking do caminho mais semelhante para a i-ésima tarefa buscada (Tb_i) e n é o número de tarefas buscadas que obtiveram um caminho recomendado aceito.

Para chegar a esta fórmula, foi considerado que o ranking da primeira tarefa do caminho mais semelhante é $r(Tb_i)$ e o dos demais $(S(Tb_i) - 1)$ elementos recomendados é 1, para um total de $S(Tb_i)$ tarefas recomendadas. Os dois termos que representam os rankings são somados e divididos pelo total de tarefas recomendadas. Desta forma, caminhos que obtiveram a mesma posição no ranking, mas com uma semelhança maior, recebem um valor de M(r) menor.

Para o exemplo anterior, o resultado de M(r) é:

$$M(r) = \frac{11 + (2 - 1) + 8 + (4 - 1) + 4 + (1 - 1)}{2 + 4 + 1} = 3{,}85$$

Para esta dissertação, foi avaliada a construção de 8 workflows, e os resultados dos requisitos funcionais estão expostos na Tabela 5.5.

Tabela 5.5 - Resultado da avaliação funcional.

	VisComplete				Abordagem Proposta			
	Tarefas Inseridas Manualmente	Total de Tarefas Recomendadas	Percentual de Tarefas Recomendadas	Média do Ranking	Tarefas Inseridas Manualmente	Total de Tarefas Recomendadas	Percentual de Tarefas Recomendadas	Média do Ranking
wf1	11	7	39%	1	11	7	39%	1
wf2	3	5	63%	1,8	3	5	63%	3,6
wf3	6	3	33%	1	6	3	33%	1
wf4	8	3	27%	1	8	3	27%	1
wf5	5	8	62%	1,25	7	6	46%	1
wf6	5	7	58%	1,71	5	7	58%	1,28
wf7	6	7	54%	3,85	6	7	54%	3,42
wf8	5	5	50%	2,4	5	5	50%	2,6
Total	49	45	48%	1,75	51	43	46%	1,86

Os resultados mostram que ambas as abordagens obtiveram resultados muito próximos, reduzindo em quase 50% a necessidade de se inserir manualmente uma atividade durante a criação de um novo workflow. A média da média dos rankings indica que, para ambas as abordagens, a recomendação ideal está normalmente no primeiro ou segundo item da lista de recomendações.

5.4.3. Casos Particulares de Recomendação

Durante a realização da avaliação, um caso particular de workflow se destacou, mostrando um comportamento bem diferente nas duas abordagens. Este caso ocorre quando há uma tarefa-ruído, tarefa que não está presente em nenhum caminho padrão, entre duas tarefas fortemente conectadas.

Para ilustrar esse caso, considere novamente a Figura 5-8. Ao se adicionar a tarefa vtkStripper não houve nenhuma recomendação, isso ocorreu pois vtkStripper é um ruído, ou seja, é uma tarefa muito pouco utilizada e por isso não aparece na base de caminhos padrão. Como o VisComplete utiliza todos os workflows criados para recomendar, é possível afirmar que a tarefa vtkStripper não aparece em nenhum workflow previamente criado. Ao continuar a recomendação, Figura 5-8.c, o caminho mais semelhante calculado pelo VisComplete para a tarefa buscada vtkProbeFilter está

na oitava posição. Isto ocorreu porque como vtkStripper não está em nenhum caminho conhecido, todo caminho já criado é descartado na busca, e a sinergia entre as tarefas vtkPolyDataNormals e vtkPolyDataMapper é perdida. Este tipo de problema não ocorre na abordagem proposta, pois a Etapa de Limpeza da Fase de Consulta e Recomendação elimina o ruído, permitindo utilizar todo o caminho criado para realizar a busca. A Figura 5-9 exibe a recomendação do VisComplete quando a tarefa ruído vtkStripper é descartada. Observe que agora o caminho mais semelhante passa a ser o primeiro da lista de recomendação, como já ocorria na recomendação gerada por esta abordagem.

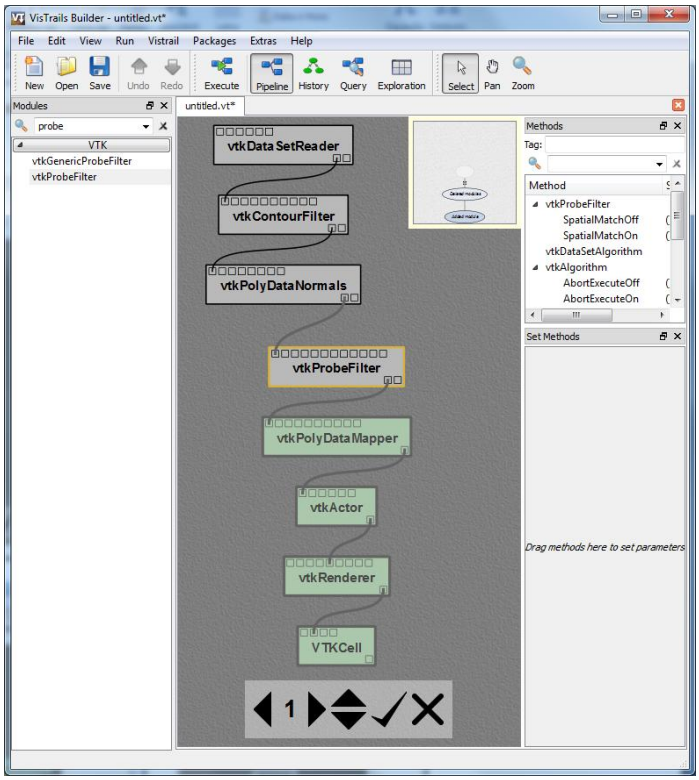


Figura 5-9 - Caso particular de recomendação.

Apenas um dos testes realizados nesta avaliação acarretou no caso exposto anteriormente, acarretando em uma perda significativa na precisão da recomendação. O tratamento destes casos, como o apresentado neste exemplo, torna um grande diferencial a presença de artifícios que resolvam este tipo de problema.

Outra avaliação realizada, que não está descrita no capítulo, foi o tempo necessário para se obter uma recomendação a partir do momento que a busca é enviada. O resultado foi omitido, pois para ambas as abordagens o tempo necessário foi mínimo, com o pior caso demorando apenas três segundos.

5.5. Considerações Finais

Este capítulo apresentou como foi feita a implementação e a avaliação do protótipo desta dissertação. Também avaliou o VisComplete, permitindo comparar as duas abordagens. A ênfase da avaliação foi no desempenho da fase de preparação e na precisão da fase de consulta e recomendação.

Os resultados desta avaliação para a abordagem proposta são promissores, porém há a necessidade de melhorar a usabilidade do protótipo para que ele possa ser utilizado efetivamente.

Outra vantagem da abordagem proposta é a utilização de um algoritmo aberto conhecido, pois caso haja uma melhora ou apareça um novo algoritmo de mineração sequencial, a abordagem pode fazer o uso desse algoritmo e indiretamente ser beneficiada.

O próximo capítulo apresenta as contribuições, limitações e trabalhos futuros desta dissertação.

Capítulo 6 - Conclusões

6.1. Contribuições

Com o intuito de atenuar o cenário atual de experimentos científicos, onde se utiliza programas em *pipelines* baseados em *scripts* de programação, sistemas de gerência de workflows científicos (SGWfC) passaram a ser utilizados. Porém, com o crescimento desses sistemas, eles estão se tornando cada vez mais complexos e disponibilizando mais funcionalidades aos usuários, e o encadeamento dos programas e serviços disponíveis está cada vez mais difícil. Atualmente, há uma dependência exclusiva da capacidade individual dos pesquisadores no encadeamento dos programas necessários para a execução de experimentos. A motivação deste trabalho, portanto, está na possibilidade de melhorar o cenário atual, facilitando e agilizando o desenvolvimento de novos workflows através de recomendação.

Com base nesta motivação, o objetivo desta dissertação foi desenvolver um sistema de recomendação para workflows que permita sugerir as combinações de programas mais frequentes, de forma a acelerar o processo de construção de workflows e melhorar a qualidade dos workflows desenvolvidos, melhorando o desempenho e a precisão das recomendações frente às abordagens existentes disponíveis. Esta reutilização utiliza informações de workflows previamente desenvolvidos para auxiliar no desenvolvimento de novos workflows, sugerindo como os programas e serviços devem ser encadeados.

A abordagem desenvolvida é a primeira a utilizar a técnica de mineração de dados chamada mineração sequencial para gerar as recomendações em workflows. Esse tipo de mineração é amplamente utilizado para gerar recomendações no domínio de comércio eletrônico e no decorrer da dissertação foi feita uma analogia entre os domínios para que fosse possível utilizar esta técnica de mineração.

A abordagem foi dividida em duas etapas: **preparação**, responsável por extrair os caminhos dos workflows por domínio de aplicação, atribuir peso aos workflows mais utilizados e aplicar o algoritmo de mineração sequencial para extrair as regras de associação entre as tarefas dos workflows; e **consulta e recomendação**, que como o próprio nome já diz, é responsável por receber a consulta do usuário, tratá-la e retornar as recomendações.

Um protótipo foi implementado para permitir a avaliação da abordagem e comparar os resultados com um sistema disponível, o VisComplete. A ênfase da avaliação foi no desempenho da fase de preparação e na precisão da fase de consulta e recomendação. As métricas utilizadas para avaliar o desempenho foram tempo e gasto de memória RAM, e os itens avaliados foram as etapas envolvidas na fase de preparação e o carregamento das bases de caminhos máximos durante a fase de consulta e recomendação. Para a avaliação da precisão, os requisitos foram o percentual de tarefas que foram recomendadas por workflow e a média dos rankings das recomendações, onde o ranking é a posição na lista de recomendações que está a recomendação ideal.

Os resultados da avaliação para a abordagem proposta são promissores. Em relação ao desempenho, os resultados mostraram que a abordagem foi superior ao VisComplete, principalmente para grandes quantidades de workflows. Em relação à precisão, ambas as abordagens obtiveram valores muito próximos, reduzindo em quase 50% o trabalho de construção de novos workflows e mantendo uma média de menos de dois para o ranking das recomendações.

Como a abordagem proposta utiliza uma técnica de mineração de dados conhecida, mineração sequencial, caso haja um aperfeiçoamento do algoritmo que

implementa esta técnica, a abordagem é beneficiada indiretamente, melhorando ainda mais seu desempenho.

Com a implementação do protótipo foi possível detectar uma característica importante desta abordagem, que é a capacidade de gerar recomendações de forma precisa mesmo quando há tarefas-ruído em uma consulta.

6.2. Limitações

Como o protótipo implementado neste trabalho tem por objetivo avaliar apenas o desempenho e a precisão das recomendações, não foi desenvolvida uma interface gráfica, não sendo possível avaliar a usabilidade da ferramenta.

A utilização de uma base de workflows com apenas um domínio também prejudicou a avaliação da abordagem, impossibilitando a avaliação de um diferencial, que é a separação das bases de recomendação por domínio. Provavelmente os resultados seriam melhores caso houvesse essa separação. Também não foi possível avaliar a aplicação de pesos aos workflows, visto que não tínhamos informações sobre o número de execuções dos workflows presentes na base.

6.3. Trabalhos futuros

Uma avaliação importante que não foi possível realizar nesta dissertação é a utilização do sistema de recomendação em um ambiente de desenvolvimento real para avaliar sua usabilidade, fato que só seria possível após a integração deste sistema a um SGWfC ou caso fosse desenvolvida uma interface gráfica própria.

Todos os testes realizados utilizaram exemplos do SGWfC VisTrails. A adaptação do protótipo para utilização com outros SGWfC em conjunto com a criação de uma analogia entre as tarefas de cada sistema, permitiria ampliar o escopo e a

precisão das recomendações, gerando uma ferramenta de recomendação para todos os SGWfC.

Referências Bibliográficas

AGARWAL, R.C., AGGARWAL, C., PRASAD, V.V.V., 2000, "A Tree Projection Algorithm for Generation of Frequent Itemsets", *Journal of Parallel and Distributed Computing*, v. 61, n. 3 (Mar), pp. 350 - 371.

AGGARWAL, C., WOLF, J., WU, K., AND YU, P, 1999, "Horting hatches an egg: A new graph-theoretic approach to collaborative filtering". In: *ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 201 - 212, San Diego, California, USA.

AGRAWAL, R., IMIELINSKI, T., SWAMI, A. , 1993, "Mining Association Rules between Sets of Items in Large Databases". In: *ACM SIGMOD international conference on Management of data*, pp. 207-216, Washington, D.C., USA.

AGRAWAL, R., IMIELINSKI, T., SWAMI, A. , 1994, "Fast Algorithms for Mining Association Rules". In: *VLDB Conference*, pp. 487-499, San Francisco, California, USA.

AGRAWAL, R., RAMAKRISHNAN, S., 1995, "Mining Sequential Patterns". In: *Proceedings of the Eleventh International Conference on Data Engineering*, pp.3-14, Taiwan.

AGRAWAL, R.S.R., 1996, "Mining sequential patterns: Generalizations and performance improvements". In: *Proceedings of the 5th International Conference on Extending Database Technology*, pp. 3-17, France.

ALTINTAS, I., BERKLEY, C., JAEGER, E., JONES, M., LUDASCHER, B., MOCK, S., 2004, "Kepler: an extensible system for design and execution of scientific workflows". In: *International Conference on Scientific and Statistical Database Management*, pp. 423-424, Santorini, Greece.

BALABANOVIC, M.A.S., Y., 1997, "FAB: Content-based collaborative recommendation", v. 40, n. 3, pp. 66 - 72.

BASU, C., HIRSH, H., AND COHEN, W., 1998, "Recommendation as classification: Using social and content-based information in recommendation". In: *Workshop on Recommender Systems*, pp. 11–15, Reston, Virginia, USA.

BEEFERMAN, D.A.B., A., 2000, "Agglomerative clustering of a search engine query log". In: *ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 407–415, New York, USA.

BILLSUS, D.A.P., M. J., 1998, "Learning collaborative information filters". In: *Proceedings of International Conference on Machine Learning*, pp. 46–54, Madison, Wisconsin, USA.

BREESE, J.S., HECKERMAN, D., AND KADIE, C. , 1998, "Empirical Analysis of Predictive Algorithms for Collaborative Filtering". In: *14th Conference on Uncertainty in Artifcial Intelligence*, pp. 43-52, Madison, Wisconsin, USA.

BRIN, S.M., R. TSUR, S., 1997, "Dynamic itemset counting and implication rules for market basket data". In: *International Conference on Management of Data*, pp. 255 - 264, Tucson, Arizona, USA.

CALLAHAN, S.P., FREIRE, J., SANTOS, E., SCHEIDEGGER, C. E., SILVA, C. T., VO, H. T., 2006, "VisTrails: visualization meets data management". In: *ACM SIGMOD*, pp. 745-747, Chicago, IL, USA.

CRUZ, S.M.S.D.E.S.F.T.C.V.R.C.D.T.D., A.; CAMPOS, MARIA LUIZA; MATTOSO, M., 2008a, "OrthoSearch: A Scientific Workflow Approach to Detect Homologies on Protozoans". In: *23rd ACM Symposium on Applied Computing*, v. ii, pp. 1281 - 1286, Fortaleza, CE, Brasil.

CRUZ, S.S., MATTOS, A., SILVA, F., RUBERG, N., MATTOSO, M. L., 2008b, *Gerência de Workflows Científicos: Uma Análise Crítica No Contexto da Bioinformática*, PESC/COPPE/UFRJ, Rio de Janeiro, Brasil.

EZEIFE, C.I.Y., LU, 2005a, "MiningWeb Log Sequential Patterns with Position Coded Pre-Order LinkedWAP-Tree", *Journal of Data Mining and Knowledge Discovery*, v. 10, pp. 5 - 38.

EZEIFE, C.I.Y., LU YI, LIU, 2005b, "PLWAP sequential mining: open source code". In: *1st International Workshop on Open Source Data Mining: Frequent Pattern Mining Implementations* pp. 26-35, Chicago, Illinois.

FAYYAD, U., PIATETSKY-SHAPIRO, G., SMYTH, P., 1996b, "The KDD Process for Extracting Useful Knowledge from Volumes of Data", v. 39, n. 11, pp. 27-34.

FAYYAD, U.M., PIATETSKY-SHAPIRO, G., SMYTH, P., AND UTHURUSAMY, R., EDS., 1996a, "Data Mining and Knowledge Discovery in Databases", **Communications of the ACM**, v.39, n.11 (Nov), pp. 24-26.

GOLDBERG, D., NICHOLS, D., OKI, B. M., AND TERRY, D., 1992, "Using collaborative filtering to weave an information tapestry", v. 35, n. 12, pp. 61–70.

GOLDSCHMIDT, R., PASSOS, E., 2005, *Data Mining: Um Guia Prático*, 1 ed.

HAND, D., MANNILA, H., SMYTH, P., 2001, *Principles of Data Mining*, 1 ed. Massachusetts.

HAND, D.J., 1981, *Discrimination and Classification*, John Wiley & Sons Inc.

HERLOCKER, J., KONSTAN, J., BORCHERS, A., AND RIEDL, J., 1999, "An algorithm framework for performing collaborative filtering". In: *SIGIR*, pp. 77–87, New York, USA.

HILL, W., STEAD, L., ROSENSTEIN, M., AND FURNAS, G., 1995, "Recommending and evaluating choices in a virtual community of use". In: *Proceedings of the SIGCHI conference on Human factors in computing systems*, pp. 194-201, Denver, Colorado, USA.

HOLT, D.C., M., 1999, "Efficient mining of association rules in text databases". In: *Conference on Information and Knowledge Management*, pp. 234 - 242, Kansas City, Missouri, USA.

HULL, D., WOLSTENCROFT, K., STEVENS, R., GOBLE, C., POCOCK, M. R., LI, P., OINN, T., 2006, "Taverna: a tool for building and running workflows of services", v. 34, n. Web Server issue, pp. 729-732.

JAIN, A.K., MURTY, M. N., FLYNN, P. J., 1999, "Data Clustering: A Review". In: *ACM Computing Surveys*, v. 31, n. 3, pp. 264-323.

JIAWEI, H., JIAN, P., YIWEN, Y., 2000, "Mining frequent patterns without candidate generation". In: *ACM SIGMOD international conference on Management of data*, pp. 1-12, Dallas, Texas, USA.

JOSEPH A. KONSTAN, B.N.M., DAVID MALTZ, JONATHAN L. HERLOCKER, LEE R. GORDON, AND JOHN RIEDL, 1997, "GroupLens: Applying Collaborative Filtering to Usenet news", v. 40, n. 3, pp. 77-87.

JOSHI, M.K., G. KUMAR, V., 2001, "A Universal Formulation of Sequential Patterns". In: *Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1-21, San Francisco, USA.

KITTS, B., FREED, D., AND VRIEZE, M., 2000, "Cross-sell: A fast promotion-tunable customer–item recommendation method based on conditional independent probabilities". In: *ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 437–446, New York.

KONSTAN JOSEPH A., B.N.M., DAVID MALTZ, JONATHAN L. HERLOCKER, LEE R. GORDON, AND JOHN RIEDL, 1997, "GroupLens: Applying collaborative filtering to Usenet news", v. 40, n. 3, pp. 77-87.

KOOP, D.S., C.E. CALLAHAN, S.P. FREIRE, J. SILVA, C.T., 2008, "VisComplete: automating suggestions for visualization pipelines", v. 14, n. 6, pp. 8.

LEMOS, M., 2004, *Workflow para Bioinformática*, Tese de D.Sc., Programa de Pós-graduação em Informática, PUC-Rio, Rio de Janeiro, RJ, Brasil.

MATTOSO, M.L.Q.W., C. M. L. ; TRAVASSOS, G. H. ; BRAGANHOLO, V. ; MURTA, L. G. P. ; OGASAWARA, E. ; OLIVEIRA, F. T. ; MARTINHO, W., 2009, "Desafios no apoio à composição de experimentos científicos em larga escala", In: *SEMISH - CSBC*, Bento Gonçalves, RS, Brasil.

MOBASHER, B., COOLEY, R., AND SRIVASTAVA, J., 2000, "Automatic personalization based on web usage mining", v. 43, n. 8, pp. 142–151.

MUKUND, D., GEORGE, K., 2004, "Item-based top-N recommendation algorithms", ***ACM Transactions on Information Systems (TOIS)*** v.22 n.1 (Jan), p.143-177.

OLIVEIRA, F.T.M., L. G. P.; WERBER, C. M. L.; MATTOSO, M. L. Q., 2008, "Using Provenance to Improve Workflow Design". In: *International Provenance and Annotation Workshop*, v. 5272, pp. 136 - 143, Salt Lake City, USA.

PARK, J., CHEN. M., YU, P., 1995, "An Effective Hash-based Algorithm for Mining Association Rules". In: *ACM SIGMOD Conference on Management of Data*, v.24 , n. 2, pp. 175-186, San Jose, California, USA.

PEI, J.H., J. MORTAZAVI-ASL, B. ZHU, H., 2000, "Mining access patterns efficiently from web logs". In: *Proceedings of the 4th Pacific-Asia Conference on Knowledge Discovery and Data Mining*, Current Issues and New Applications, pp. 396-407, Kyoto, Japan.

RASURE, K., 1994, "The Khoros Application Development Environment". In: *Experimental Environments for Computer Vision and Image Processing*, v. 11, *Machine Perception Artificial Intelligence,* World Scientific Publishing, pp. 1-32.

RESNICK, P., IACOVOU, N., SUCHAK, M., BERGSTROM, P., AND RIEDL, J., 1994, "GroupLens: An open architecture for collaborative filtering of netnews". In: *CSCW*, pp. 175-186, Chapel Hill, North Carolina, USA.

RESNICK, P.A.V., H. R., 1997, "Recommendation systems", ***Communications of the ACM*** v. 40, n. 3 (Mar), pp. 56-58.

SARWAR, B., KARYPIS, G., KONSTAN, J., AND RIEDL, J., 2000, "Analysis of recommendation algorithms for e-commerce". In: *ACM conference on Electronic commerce*, pp. 158 - 167 Minneapolis, Minnesota, United States.

SCHAFER, J., KONSTAN, J., AND RIEDL, J., 1999, "Recommender systems in e-commerce". In: *ACM conference on Electronic commerce*, pp. 158 - 166, Colorado, USA.

SCHROEDER, W.M., K. LORENSEN, B., 2003, *The Visualization Toolkit: An Object Oriented Approach to 3D Graphics*, 3 ed, New York, Kitware.

SHARDANAND, U.A., P., 1995, "Social information filtering: Algorithms for automating “word of mouth"", *ACM*, New York.

SILVA, W.P.S., C. M.; CAVALCANTI, C. G.; SILVA, D. D.; SOARES, I. B.; OLIVEIRA, J. A.; SILVA, C. D., 2004, "LAB Fit Ajuste de Curvas: Um Software em Protuguês para Tratamento de Dados Experimentais", v. 26, n. 4.

SOARES, J.D.A., 2007, *PRÉ-PROCESSAMENTO EM MINERAÇÃO DE DADOS: UM ESTUDO COMPARATIVO EM COMPLEMENTAÇÃO*, Tese D.Sc., COPPE, UFRJ, Rio de Janeiro.

SOWIZRAL, H.R., K. DEERING, M., 1998, *The Java 3D API Specification*.

SUMPTER, R.M., 1994, "White Paper on Data Management", Washington.

TAYLOR, I.J., DEELMAN, E., GANNON, D. B., SHIELDS, M., 2006, *Workflows for e-Science: Scientific Workflows for Grids*, 1 ed., Springer.

TELEA, A.J.V.W., 1999, "VISSION: An Object Oriented Dataflow System for Simulation and Visualization". In: *Proceedings of IEEE VisSym*, Vienna, Austria.

TELEA, A.J.V.W., 2000, "SmartLink: An Agent for Supporting Dataflow Application Construction", *Springer*, pp. 189-198, Vienna.

TERVEEN, L., HILL, W., AMENTO, B., MCDONALD, D., AND CRETER, J., 1997, "PHOAKS: A system for sharing recommendations", v. 40, n. 3, pp. 59-62.

UPSON, C.F., T., 1989, "The Application Visualization System: A Computational Environment for Scientific Visualization", *IEEE Computer Graphics and Applications*, v. 9, n. 4 (Julho 1989), pp. 30 - 42.

WEISS , S.M., KULIKOWSKI, C. A., 1991, *Computer systems that learn: classification and prediction methods from statistics, neural nets, machine learning, and expert systems*, Morgan Kaufmann Publishers Inc.

WFMC, D.H., *Workflow Management Coalition Terminology &* Glossary: WFMC, 2006. Disponível em: http://www.wfmc.org/index.php?option=com_docman & task= doc_download&gid=93&Itemid=72 , Acesso em: 10 mai. 2010.

ZAKI, J.P., S. OGIHARA, M. LI, WEI 1997, "Parallel Algorithms for Discovery of Association Rules", v. 2, n. 4, pp. 343 - 373, Dezembro 1997.

ZHANG, S., ZHANG, C., YANG, Q., 2003, "Data Preparation for Data Mining", v. 17, n. 5-6, pp. 375-381, Maio-Junho.

ZHAO, Y., DOBSON, J., FOSTER, I., MOREAU, L., WILDE, M., 2005, "A notation and system for expressing and executing cleanly typed workflows on messy scientific data", v. 34, n. 3, pp. 37-43.

ZHUGE, H., 2003, "Component-based Workflow System Development", v. 35, n. 4, pp. 517 - 536, July.

Apêndice A – Arquivo de Entrada para o Algoritmo PLWAP

Este apêndice exibe parte do arquivo de entrada para o algoritmo PLWAP. Cada linha contém o identificador do caminho, o número de tarefas do caminho e a sequência de tarefas que representa o caminho.

```
1     6     5647  6710  6380  5801  5836  6184
2     3     5647  6710  6548
3     4     5646  5650  5627  6184
4     6     5647  6710  6380  5801  5836  6175
5     6     5647  6710  6380  5801  5836  6175
6     6     5647  6710  6380  5801  5836  6175
7     6     6710  6380  5801  5756  6139  6534
8     3     6710  6380  6704
9     3     6380  5804  5906
10    2     6380  6704
11    6     6710  6380  5801  5756  6139  6534
12    3     6710  6380  6704
13    7     6710  6380  5804  5906  5756  6139  6534
14    3     6710  6380  6704
15    6     5647  6710  6380  5801  5836  6175
16    3     5647  6710  6548
17    4     5646  5650  5627  6175
18    6     5647  6710  6380  5801  5836  6175
19    3     5647  6710  6548
20    4     5646  5650  5627  6175
21    6     5647  6710  6380  5801  5836  6175
22    4     5646  5650  5627  6175
23    5     5647  6710  6380  5801  6175
24    6     5647  6710  6380  5801  5836  6184
25    3     5647  6710  6548
26    4     5646  5650  5627  6184
27    6     5647  6710  6380  5801  5836  6175
28    6     5647  6710  6380  5801  5836  6175
29    6     5647  6710  6380  5801  5836  6175
30    7     5647  6710  6380  5801  5756  6139  6534
31    4     5647  6710  6380  6704
32    8     5647  6710  6380  5804  5906  5756  6139  6534
33    4     5647  6710  6380  6704
34    5     5647  6710  6380  5801  6175
35    4     5646  5650  5627  6175
36    5     5647  6710  6380  5801  6175
37    6     5647  6710  6380  5801  5836  6184
38    3     5647  6710  6548
39    4     5646  5650  5627  6184
40    6     5647  6710  6380  5801  5836  6175
41    6     5647  6710  6380  5801  5836  6175
42    7     5647  6710  6380  5801  5756  6139  6534
43    4     5647  6710  6380  6704
44    8     5647  6710  6380  5804  5906  5756  6139  6534
45    4     5647  6710  6380  6704
46    6     5647  6710  6380  5801  5836  6175
47    4     5646  5650  5627  6175
48    7     5647  6710  6380  5801  5756  6139  6534
49    4     5647  6710  6380  6704
50    8     5647  6710  6380  5804  5906  5756  6139  6534
```

Apêndice B – Arquivo Gerado pelo Algoritmo PLWAP

Este apêndice apresenta parte do arquivo de saída gerado pelo algoritmo PLWAP. Cada linha contém um caminho padrão com seu respectivo suporte.

```
5616  0.305825
5626  0.380097
5626  5616  0.297087
5627  0.357282
5627  5616  0.297573
5627  5626  0.319903
5627  5626  5616  0.297087
5641  0.100485
5646  0.455825
5646  5616  0.297573
5646  5626  0.319903
5646  5626  5616  0.297087
5646  5627  0.357282
5646  5627  5616  0.297573
5646  5627  5626  0.319903
5646  5627  5626  5616  0.297087
5646  5641  0.0985437
5646  5650  0.357282
5646  5650  5616  0.297573
5646  5650  5626  0.319903
5646  5650  5626  5616  0.297087
5646  5650  5627  0.357282
5646  5650  5627  5616  0.297573
5646  5650  5627  5626  0.319903
5646  5650  5627  5626  5616  0.297087
5647  0.528641
5647  5626  0.0582524
5647  5801  0.207767
5647  5801  5836  0.0708738
5647  5801  5836  6175  0.05
5647  5801  6175  0.0708738
5647  5801  6178  0.0558252
5647  5804  0.136893
5647  5804  5958  0.0543689
5647  5804  6178  0.0558252
5647  5836  0.0762136
5647  5836  6175  0.05
5647  5958  0.0543689
5647  6175  0.0708738
5647  6178  0.116505
5647  6380  0.42767
5647  6380  5801  0.193204
5647  6380  5801  5836  0.0684466
5647  6380  5801  5836  6175  0.05
5647  6380  5801  6175  0.0708738
5647  6380  5804  0.136893
```

Apêndice C – Arquivo Resultante de uma Recomendação

Este apêndice exibe um arquivo gerado a partir de uma consulta. Na primeira linha, é apresentada a consulta de entrada e cada linha restante representa uma recomendação, com sua respectiva frequência.

```
Consulta: ['1228', '2291', '1759']
Resultados:
['1228', '2291', '1385', '1759'] Frequência: 0.117123
['1228', '2291', '1961', '1385', '1759'] Frequência: 0.117123
['1228', '2291', '2285'] Frequência: 0.115132
['1228', '2291', '1961', '2285'] Frequência: 0.114471
['1228', '2291', '2176'] Frequência: 0.0694784
['1228', '2291', '2177'] Frequência: 0.0614743
['1228', '2291', '1765'] Frequência: 0.056611
['1228', '2291', '1570', '1759'] Frequência: 0.0487274
['1228', '2291', '1961', '1570', '1759'] Frequência: 0.0479578
['1228', '2291', '1961', '1765'] Frequência: 0.0394553
['1228', '2291', '1961', '2177'] Frequência: 0.0392119
['1228', '2291', '1961', '2176'] Frequência: 0.0386903
['1228', '2291', '1454', '1759'] Frequência: 0.0343718
['1228', '2291', '1961', '1454', '1759'] Frequência: 0.0343718
['1228', '2291', '1756'] Frequência: 0.0343208
['1228', '2291', '2129'] Frequência: 0.0336347
['1228', '2291', '1961', '1756'] Frequência: 0.0335466
['1228', '2291', '1417', '1759'] Frequência: 0.0330389
['1228', '2291', '1961', '1417', '1759'] Frequência: 0.0328071
```