Escalabilidade e Eficiência em Mineração de Dados de Aplicações Internet*

Wagner Meira Jr.[1] **, Renato Ferreira**[1] **, Dorgival Guedes**[1]

[1]Departamento de Ciência da Computação
Universidade Federal de Minas Gerais
31.270-010 – Belo Horizonte – MG – Brasil

Resumo. *A Internet foi muito além de um artefato tecnológico, passando a ser um instrumento crescente de interação social. Essas interações são usualmente complexas e difíceis de analisar automaticamente, demandando o desenvolvimento de novas técnicas de mineração de dados que se adaptem às peculiaridades dos cenários de aplicação. Por sua vez, essas novas técnicas, à semelhança de outras técnicas de mineração de dados, são intensivas em termos de computação e de entrada e saída, motivando a pesquisa e o desenvolvimento de novos paradigmas, ambientes de programação e algoritmos paralelos que executem essas tarefas com escalabilidade e eficiência. Os resultados descritos no artigo apontam não apenas a pertinência do desenvolvimento dessas novas técnicas, como também a sua paralelização. Mais ainda, permitem identificar três grupos de desafios de pesquisa em Ciência da Computação, assim como as suas demandas mais relevantes.*

1. Contexto

Um dos aspectos mais interessantes da Internet é que ela tem, de forma crescente, representado mais do que o artefato tecnológico em si para que foi concebida. O uso da Internet não só retirou barreiras físicas inerentes às formas de interação que já existiam, como também habilitou o aparecimento de novas formas de interação social, muitas vezes complexas e inéditas. Por outro lado, tarefas como extração automatizada de informação, caracterização da natureza da interação e detecção de tendências são ainda problemas em aberto, seja pela complexidade da própria tarefa, seja pelo volume de dados, seja pela sua evolução constante. Este artigo discute essas demandas e apresenta algumas soluções baseadas em técnicas de mineração de dados, as quais, para satisfazerem requisitos de escalabilidade e eficiência, exploram a computação paralela e distribuída.

Mineração de dados, de acordo com uma definição comumente adotada, é a extração automatizada ou semi-automatizada de padrões interessantes a partir de grandes bases de dados. A definição de quais padrões são interessantes é subjetiva e depende da tarefa sendo realizada e do analista que avalia os padrões sendo minerados. Na prática, mineração de dados é um conjunto de técnicas desenvolvidas ao longo de várias décadas em várias áreas do conhecimento como Estatística, Inteligência Artificial e Recuperação de Informação. Além dos avanços em termos de técnicas, a capilarização dos recursos de tecnologia da informação e a maior capacidade de processamento e armazenamento de plataformas computacionais modernas estão entre os principais fatores para o crescimento da área de Mineração de Dados como um todo.

*Este trabalho foi parcialmente financiado por CNPq, Fapemig e Finep.

Entretanto, podemos identificar várias demandas de informação associadas a aplicações Internet onde técnicas tradicionais estão aquém dessas demandas. Em particular, duas características destas demandas se mostram como grandes desafios. A primeira característica é a natureza dos padrões a serem minerados, que não apenas são mais complexos, mas sua determinação tem, em geral, maior custo computacional. No caso de aplicações Internet, o fato dos padrões refletirem interações sociais e a evolução implícita desses padrões são desafios claros. O segundo problema é relacionado à escalabilidade das técnicas, uma vez que muitas delas já possuem uma complexidade exponencial para padrões mais simples, e o custo tende a aumentar para padrões mais complexos. A seguir discutimos quatro cenários de aplicações Internet e algumas das técnicas de mineração de dados que vêm sendo desenvolvidas nesses contextos.

Um dos exemplos mais simples e poderosos da Internet como instrumento de interação social são as mensagens eletrônicas (*e-mails*). Apesar de ser um modelo transposto do mundo real, essas mensagens basicamente mudaram fundamentalmente as interações entre organizações e pessoas, impondo uma agilidade de comunicação sem precedentes, ao mesmo tempo que permitem uma redução de custos significativa. Por outro lado, a facilidade de envio de mensagens permitiu o surgimento de uma das grandes ameaças ao uso da Internet como canal de comunicação: as mensagens não solicitadas ou spams. Avaliações recentes indicam que uma fração significativa das mensagens que hoje trafegam na Internet são não solicitadas. As finalidades dessas mensagens são as mais variadas, desde divulgação comercial até disseminação de programas maliciosos, passando por lendas urbanas. A detecção dessas mensagens se mostra um desafio não apenas em razão do seu volume, que já é grande e continua crescendo, mas também pela mudança contínua das estratégias usadas pelos *spammers*, de forma a contornar os mecanismos de detecção existentes. A exploração de técnicas de mineração de dados para o entendimento da natureza dos *spams* e o desenvolvimento de mecanismos mais eficazes de detecção é a nossa perspectiva de atuação. Um grande número de técnicas para detecção dessas mensagens têm sido propostas, incluindo técnicas de mineração de dados [6, 9]. A nossa abordagem se baseia em considerar evidências heterogêneas e combiná-las de forma mais efetiva para detectar essas mensagens. Resultados preliminares desta abordagem são muito promissores [31, 30] e discutidos neste artigo juntamente com resultados mais recentes.

Outro exemplo de aplicação da Internet de grande impacto são as bibliotecas digitais, que também exploraram a facilidade de acesso provida pela Internet. A construção de bibliotecas digitais foi uma das aplicações iniciais da World Wide Web e desde então os mecanismos e algoritmos associados a estas bibliotecas vêm constantemente se aperfeiçoando. Da mesma forma, podemos observar uma crescente população de usuários e um conteúdo cada vez mais diverso e com maior cobertura temporal, tendo em vista não apenas a incorporação de novos documentos, mas também a digitalização de documentos mais antigos. Cabe ressaltar que a crescente digitalização de documentos de toda ordem criou um acervo gigantesco de documentos, no qual a sua recuperação se mostra uma tarefa difícil. O desenvolvimento de técnicas efetivas de classificação automática de textos ainda é uma demanda, apesar de décadas de pesquisa no assunto [2]. Trabalhos anteriores dos quais participamos aplicaram com sucesso técnicas de mineração de dados no contexto de recuperação da informação [24]. Entretanto, esse crescimento abrupto das bibliotecas digitais trouxe vários desafios. Um primeiro desafio é a extração de carac-

terísticas a partir dos documentos, como por exemplo a identificação de elementos como autores e referências bibliográficas, as quais são elementos fundamentais para, entre outras finalidades, analisar a relevância e significância dos documentos. Um segundo desafio é a classificação automática de documentos, que é fundamental para a sua localização e contextualização, auxiliando usuários a determinar documentos que sejam mais adequados a suas demandas. Um desafio maior ainda é como as informações fundamentais para a extração e a classificação evoluem ao longo do tempo, e, mais ainda, como desenvolver mecanismos que sejam adaptativos a essa evolução. Neste artigo enfocamos a questão de classificação de textos, onde temos por objetivo gerar classificadores que sejam capazes de lidar com múltiplas evidências e critérios de relevância, enquanto mantém a sua eficiência computacional. Até o momento, os resultados obtidos [28, 30, 29] indicam que a nossa estratégia supera o estado-da-arte em termos de precisão, como discutido neste artigo.

Um outro exemplo de aplicação da Internet de grande impacto na sociedade é o comércio eletrônico, que no nosso contexto é o conjunto de ferramentas e tecnologias que suportam a realização de transações comerciais na Internet. A implementação de serviços de comércio eletrônico coloca várias demandas em termos de correção das transações, escalabilidade, segurança das informações e robustez, as quais vêm sendo gradativamente contempladas por vários tipos de desenvolvimento tecnológico. Mas a natureza dos recursos de interação e a mudança do ambiente onde ocorrem as transações comerciais levam a alterações significativas no perfil de comportamento dos clientes e como eles interagem com o sistema e com outros participantes. Um cenário onde essas alterações são claras são os mecanismos de precificação dinâmica, como leilões e pregões. Esses mecanismos já são extremamente populares e trazem uma série de desafios do ponto de vista de pesquisa em Ciência da Computação. Um desafio em particular é entender o que influencia o comportamento dos clientes e, portanto, o resultado da negociação dinâmica. Entretanto, ainda não está claro o que determina o sucesso ou não de uma venda por preço dinâmico, e quanto da dinâmica do processo está associada a razões de impulso de compra. Diversas técnicas de mineração de dados têm sido aplicadas ao contexto de caracterização do comportamento dos usuários [16], mas a correlação entre a resposta do usuário e do sistema não foi realmente estudada. A aplicação de técnicas de mineração de dados a outras modalidades de comércio eletrônico, como compras governamentais através da Internet [13], já demonstrou a viabilidade e impacto do uso destas técnicas. Entretanto, os padrões ainda são simples se considerarmos o nível de detalhamento dos dados e o potencial de informações interessantes que podemos obter. Um conceito que introduzimos para esse tipo de análise é a reatividade, ou seja, a criação de modelos a serem minerados que expressam a interdependência entre o comportamento dos usuários e dos sistemas. Este conceito já foi explorado no contexto de desempenho de serviços Web [22, 23] e temos trabalhado no sentido de estendê-lo no contexto de comércio eletrônico [21], conforme vamos discutir.

Finalmente, um último exemplo de aplicação que mimetiza a própria sociedade e suas relações são as comunidades virtuais, neste caso implementadas através de sítios de relacionamento. As comunidades virtuais são um dos grandes fenômenos da Internet nos últimos anos. Além da relevância intrínseca de tal aplicação, essas comunidades também são uma rica fonte a respeito de relacionamentos e demandas por informação e lazer dos usuários. Nestes sítios as pessoas podem indicar explicitamente o seu relacionamento com

outras pessoas “virtuais”e qualificá-lo. Mais ainda, elas podem levar a cabo discussões sobre os mais diversos assuntos, em geral com alta influência de eventos próximos temporalmente e com algum nível de relacionamento aos assuntos de interesse dos participantes. Alguns desafios associados a essas comunidades virtuais são a localização de informações relevantes a um assunto, assim como a sua natureza, por exemplo, crítica, apoio, insumo etc. Do ponto de vista de mineração de dados, a complexidade dos padrões a serem minerados, ou seja, os relacionamentos e os fatores que os explicam, herda a complexidade das próprias redes sociais, isto é, um conjunto de dados heterogêneo e multi-relacional tipicamente representado por um grafo [25]. A mineração de grafos e sub-grafos característicos dessas redes é uma tarefa desafiadora [36]. Neste artigo discutimos a mineração de relacionamentos [19], informação que pode ser útil para tarefas como resolução de entidades [3], detecção de grupos [17] e classificação de objetos. O cenário de aplicação destas técnicas são serviços Internet como Orkut, UolK e Myspace.

Em todos estes cenários temos algumas características que são comuns a essas aplicações. A primeira delas é o caráter evolutivo, onde os componentes evoluem com o tempo e os padrões minerados em um momento podem não ser válidos em outros instantes no tempo. A segunda característica é a dificuldade da extração e sistematização de informações em cada um dos cenários. Finalmente, o grande e crescente volume de dados exige técnicas e algoritmos que sejam escaláveis e eficientes. A utilização de plataformas paralelas se mostra como um caminho interessante para a escalabilidade, mas ainda persiste o desafio da eficiência. A plataforma Anthill [11] permite explorar as várias oportunidades de paralelismo da aplicação, abstraindo de detalhes de mapeamento e deixando essas decisões para tempo de execução, conforme discutido na Seção 3.1. A utilização da Anthill permitiu a paralelização de algoritmos complexos que apresentam *speedups* lineares ou quase lineares [32, 10].

Em suma, este artigo versa sobre a construção de sistemas de mineração de dados escaláveis e eficientes para um conjunto de aplicações extremamente dinâmico, com perfis de uso diversificados e que compreende enormes volumes de dados. A escolha da aplicação mineração de dados (e ferramentas de inteligência em geral) aconteceu em virtude de alguns fatores. O primeiro deles é a relevância e a demanda que existe atualmente por essas técnicas e tecnologias, tendo em vista uma quantidade crescente de dados sendo armazenada nas várias organizações e a expectativa que o conhecimento embutido em tais dados possa ser utilizado de forma produtiva. O segundo fator é o desafio técnico representado pelas técnicas em si e a obtenção de soluções escaláveis e eficientes, seja pela necessidade de manipulação de grandes volumes de dados, seja pelo perfil de execução tipicamente irregular e dinâmico dessas aplicações, exigindo técnicas e estratégias de paralelização que mascarem essas dificuldades. Finalmente, as características dessas aplicações, sua complexidade, sua irregularidade e o volume de dados a ser tratado demandam o desenvolvimento de novos mecanismos de suporte à execução das aplicações. Embora o nosso foco seja em mídias textuais, os desafios encontrados são aplicáveis, em intensidade até maior, a outras mídias como áudio, imagens e vídeo.

2. Mineração de Padrões Complexos

Conforme discutido, a mineração de padrões complexos surge como uma conseqüência natural da complexidade das aplicações e das relações que elas materializam. Para que possamos discutir em mais profundidade os desafios inerentes à mineração de padrões

complexos, vamos enfatizar o problema de classificação de textos, que no caso das aplicações discutidas, se torna um desafio pela própria ambiguidade da linguagem. Mais ainda, os avanços da área de recuperação de informação textual antecipam os desafios a serem superados em outras mídias, com a ressalva que os desafios que vamos discutir aqui também se aplicam a esses outros contextos. Desta forma, apresentamos as peculiaridades da tarefa de classificação de textos e como essas peculiaridades se manifestam em outros cenários.

Como mencionado, à medida que a quantidade de informação disponível na Internet e intranets organizacionais aumenta, cresce o interesse por ferramentas que auxiliem as pessoas a encontrar, filtrar e gerir informação. Nessa direção, classificar documentos com base em seu conteúdo é uma boa prática, pois facilita a busca pela informação desejada. Classificação de textos é a tarefa de classificar documentos com base em seu conteúdo a fim de associá-los a um ou mais assuntos (classes) pré-definidos. Historicamente, em função da introdução de documentos digitais, a automação dessa tarefa tem sido tratada como um ramo importante de pesquisa. A partir dos anos 60, uma lista de sucessos e de problemas foi reportada na literatura sobre o tema e, muito embora o entusiasmo sobre classificação tenha enfraquecido em determinada época, o rápido crescimento da Web fez reviver o interesse por classificação automática de textos [4].

Tipicamente, apenas uma pequena fração dos muitos documentos disponíveis nos vários contextos (bibliotecas digitais, emails, redes sociais) é relevante para um dado usuário. Se não soubermos o caráter do conteúdo de cada documento, é muito difícil formular buscas efetivas para analisar e extrair informações úteis de uma base de dados. Usuários necessitam de ferramentas para comparar documentos diferentes, ordená-los de acordo com sua importância, e achar padrões ou tendências entre os diversos documentos. Dessa forma, a mineração de texto [7] tem se tornado cada vez mais um tema popular e de grande relevância, tornando-se um tópico de grande interesse na área de Recuperação de Informação [15].

Além disso, a classificação automática de documentos tem aplicações em diversas áreas do conhecimento. Ela tem sido usada em assinalamento automático de rótulos a documentos, construção de diretórios de tópicos, identificação do estilo de escrita dos documentos, filtragem de SPAMs e de páginas Web com conteúdo adulto e detecção de plágio. Alguns cenários de utilização dessa técnica incluem ainda a construção de diretórios online na Web e bibliotecas digitais [8], melhorando a precisão da busca nesses ambientes, e até mesmo ajudando os usuários a interagir com máquinas de busca [27].

Devido à crescente relevância da classificação de documentos, diferentes técnicas e abordagens são exploradas. O conceito de aprendizado de máquinas é a base de muitas estratégias recentes de classificação de documentos. Nessa abordagem, os algoritmos são induzidos a “aprender” padrões com base no conteúdo de documentos previamente classificados, o que chamamos de fase de treinamento. Dessa forma, ao introduzir um novo documento de teste cuja classe é desconhecida, os padrões previamente “aprendidos” são usados, na expectativa de que seja possível deduzir corretamente a classe correspondente do documento sendo testado.

Encontramos várias técnicas que incorporam o conceito de aprendizado de máquinas, como modelos baseados em árvores de decisão, modelos probabilísticos e

vetoriais. Assim, essas abordagens incluem a classificação através de métodos como *nearest-neighbor* [18], métodos de seleção de atributos, classificação Bayesiana [20] que é uma modelagem probabilística mais simples, *support vector machines* [14] que consiste em um modelo vetorial mais complexo, e classificação baseada em associação [12, 35].

Grande parte dessas técnicas são aplicadas a cenários estáticos que não consideram a evolução temporal das coleções de documentos nem o contexto em que eles estão inseridos. Entretanto, ao longo do tempo, novas informações são criadas, novos termos são gerados, novas áreas surgem e grandes áreas são divididas em subáreas mais especializadas. O vocabulário de cada área evolui e se sobrepõe com o vocabulário de outras áreas, distorcendo as diferenças entre elas que antes eram úteis para conseguir distinguí-las. A seguir discutimos três aspectos inerentes a estes modelos: qualidade dos modelos, natureza dos dados e a necessidade de evolução de modelos e dados.

Um aspecto importante da qualidade dos modelos é a sua efetividade em termos de métricas como precisão. Uma estratégia frequentemente utilizada para a construção de modelos de mineração de dados, em particular classificadores é o emprego de modelos monolíticos, isto é, modelos que independem dos parâmetros da tarefa sendo realizada. Exemplos de tais modelos são árvores de decisão e modelos probabilísticos como redes bayesianas. No contexto das aplicações Internet aqui discutidas, os modelos monolíticos tendem a ser pouco efetivos por duas razões: alta dimensionalidade dos dados e limitações dos modelos. Para facilitar a discussão vamos fazê-la à luz do problema de detecção de spams, que pode ser facilmente generalizado para as outras aplicações.

Considerando que cada dimensão é uma palavra que ocorre em uma mensagem (ao que denominaremos genericamente termo), a dimensionalidade do problema é muito alta e afeta a efetividade dos modelos, uma vez que diferenças pequenas no espaço de busca têm que gerar saídas significativamente diferentes no modelo. Com relação às limitações do modelo, a classificação pode ser vista como um particionamento do espaço de dados do problema, onde buscamos determinar partições homogêneas em termos das classes ali representadas. Neste caso, o modelo de classificação define o “formato”da partição que é representável, o que pode limitar significativamente os modelos gerados e portanto o desempenho do classificador.

Uma solução neste caso é gerar modelos locais, ou seja, modelos induzidos pela tarefa de classificação sendo realizada. A vantagem neste caso é que o modelo é mais focado e por isso mais preciso. No contexto de mensagens eletrônicas, isto significa que termos que não ocorrem na mensagem a ser classificada não são usados. Por outro lado, o custo de geração destes modelos locais, se for alto, torna essa abordagem inviável. A nossa proposta é a classificação associativa preguiçosa [34], onde construímos modelos baseados em regras de classificação. Essa técnica foi utilizada tanto para detecção de spams [31], quanto para categorização de documentos em bibliotecas digitais [28] e detecção de sentimento em redes sociais [33], em todos os casos superando significativamente outras técnicas de classificação que representam o estado da arte.

O segundo aspecto que vamos discutir é com relação à natureza dos dados, em particular a sua heterogeneidade em termos de tipo e de informação provida. Um grande desafio para a construção de modelos de mineração de dados é a existência de dimensões heterogêneas, ou seja, dimensões cujos valores não são comparáveis. Um exemplo claro

de tais dimensões pode ser encontrado em aplicações de comércio eletrônico, em particular leilões eletrônicos. A compatibilização dessas dimensões de forma controlada é um desafio constante e tivemos sucesso em fazer isso no contexto de leilões eletrônicos, permitindo qualificar e quantificar o comportamento dos participantes do leilão [21]. Um outro cenário onde integramos múltiplas evidências no sentido de gerar modelos integrados foi na caracterização de sentimentos em redes sociais. Neste caso, consideramos não apenas o texto submetido pelos participantes, mas também o seu histórico de interação dentro da rede, permitindo superar tanto em termos de precisão, quanto de custo computacional, as técnicas que, até então, representavam o estado-da-arte [33].

A evolução constante das aplicações Internet, seja em termos dos recursos oferecidos, seja em termos do perfil de uso, resulta numa evolução dos dados e, em conseqüência, dos modelos. A evidência neste caso é que classificadores, em geral, não mantém a sua eficácia ao longo do tempo e identificar mecanismos de controlar e minorar essa degradação são o nosso objetivo. Assim, analisamos 25 anos de artigos da biblioteca da ACM e pudemos observar que não apenas a efetividade do modelo diminui à medida que o intervalo entre a sua base de treinamento e a sua utilização aumenta, como também para classificar documentos que sejam muito anteriores ao que foi utilizado para treinamento, como conseqüência da evolução dos termos ao longo do tempo e da própria semântica associada à classe que se quer prever. Neste sentido, determinamos as melhores janelas de tempo para a construção dos classificadores, e estamos estendendo os modelos de recuperação de informação para considerar esse aspecto temporal.

Estes exemplos mostram não apenas que as aplicações discutidas demandam novas técnicas e abordagens, mas também que não há uma tendência a reduzir a sua complexidade computacional, justificando a investigação de soluções que explorem a sua paralelização como será discutido a seguir.

3. Algoritmos Escaláveis e Eficientes

Nesta seção vamos discutir a paralelização de algoritmos de mineração de dados que representam as técnicas de mineração de padrões complexos e como obtemos escalabilidade e eficiência nessas implementações.

Escalabilidade e eficiência são propriedades desejáveis em sistemas de computação desde o projeto dos primeiros sistemas. Definimos escalabilidade como a capacidade do sistema de manter a qualidade dos serviços prestados a despeito do aumento da demanda imposta em termos de processamento, comunicação e acesso a dados armazenados. Eficiência, por outro lado, é o uso eficaz dos componentes do sistema, ou seja, uso que contribui realmente para a execução da tarefa atribuída ao mesmo. Além de recursos de processamento e armazenamento, podemos incluir também aspectos como a qualidade de serviço e alcance desses sistemas como critérios de eficiência. Mais importante que as definições em si, é a complementaridade desses dois conceitos: um sistema pode ser escalável e não ser eficiente ou vice-versa. Por exemplo, ao detectar um aumento da demanda, pode-se acrescentar componentes de hardware de forma a satisfazê-la, o que não significa que os componentes do sistema serão utilizados de forma eficiente. Por outro lado, um sistema pode ser eficiente sem que seja escalável, e um exemplo típico são computadores pessoais quando submetidos a tarefas de processamento intensivo. Em suma, projetar e implementar sistemas que satisfaçam essas propriedades tem sido um

desafio constante.

Uma estratégia comumente utilizada para obter tanto escalabilidade quanto eficiência é a utilização de múltiplas unidades de processamento que se comunicam e cooperam para a realização de uma tarefa. O advento de arquiteturas paralelas, o que inclui desde múltiplos processadores em uma única pastilha até a interconexão de sistemas através da Internet, propiciou um aumento sem precedentes dos recursos disponíveis para a realização de tarefas computacionais. Por outro lado, obter escalabilidade e eficiência nestes ambientes tem se mostrado uma tarefa difícil. A dificuldade em produzir sistemas escaláveis e eficientes vem não apenas da complexidade de cada um dos seus componentes, mas também das suas interações, que também são complexas.

No caso de algoritmos de mineração de dados, deve-se ressaltar três características destes algoritmos que afetam a sua paralelização e portanto a escalabilidade e eficiência. Primeiro eles tendem a ser intensivos tanto computacionalmente quanto em termos de entrada e saída. Esta característica por si só é de grande impacto, tendo em vista que os sistemas computacionais correntes tem maior foco em processamento e quando priorizam a demanda por entrada e saída, o fazem de forma estanque. Segundo, o seu comportamento tende a ser intrinsecamente irregular, tendo em vista a complexidade dos algoritmos, ou seja, não é possível prever o seu custo. Desta forma, também não é possível utilizar abordagens tradicionais, como particionamento de dados, que resultam em paralelizações mais suscetíveis a fontes de degradação de desempenho como desbalanceamento de carga. Terceiro, escalabilidade é um aspecto fundamental, tendo em vista o enorme volume de dados a ser tratado. Neste caso, o desafio vem da conjunção dos outros dois fatores, que torna a escalabilidade um desafio.

Em trabalhos anteriores [11], desenvolvemos um ambiente de programação inovador que é apresentado e discutido na Seção 3.1. O ambiente foi validado através de vários algoritmos [32, 10, 1]. Em todos os casos, há alguns requisitos comuns a esses e outros algoritmos e que expressam o desafio dessas paralelizações: obter algoritmos escaláveis em agregados de computadores e que sejam capazes de lidar com bases de dados realmente muito grandes, da ordem de terabytes ou mais, a exemplo das aplicações que discutimos na Seção 2. Discutimos essas paralelizações na Seção 3.2.

3.1. O ambiente de programação Anthill

Nesta seção descrevemos o ambiente de programação Anthill (Formigueiro), que é o nosso ambiente para desenvolvimento e execução de aplicações distribuídas escaláveis. O ambiente foi desenvolvido tendo em mente aplicações paralelas não regulares, intensivas em processamento e em entrada-e-saída de dados (E/S). Nessas aplicações, que manipulam grandes volumes de dados, estes se encontram distribuídos em várias máquinas do sistema. Mover os dados para outros nós para então serem processados é freqüentemente uma operação ineficiente. Isso é verdade especialmente porque à medida que o processamento avança, os dados resultantes tendem a ser muitas vezes menores que os dados de entrada. Dessa forma, uma alternativa interessante é levar a computação aonde o dado está, reduzindo a comunicação através da rede. O sucesso desse enfoque depende da facilidade com que a aplicação possa ser dividida em etapas que sejam passíveis de execução em nós diferentes do sistema. Cada etapa dessa forma executará parte das transformações sobre os dados, iniciando com o conjunto de dados de entrada, até que se atinja o conjunto de dados de saída.

Essa discussão indica que uma boa paralelização de qualquer aplicação nesse contexto deve considerar simultaneamente tanto paralelismo de dados quanto de tarefas. A estratégia do Anthill aplica os dois enfoques, agregando uma terceira dimensão que nos permite explorar o grau de assincronia existente entre diferentes tarefas independentes no sistema ao longo do tempo. Os benefícios dessas três dimensões combinadas nos permitem atingir *speedups* elevados experimentalmente [11].

O modelo de programação do Anthill é denominado *filter stream*. Nesse modelo, o processamento é dividido em tarefas que operam sobre os dados que fluem pelo sistema. Cada filtro implementa uma tarefa que transforma os dados segundo a necessidade da aplicação e se comunicam com outros filtros pelos canais de comunicação responsáveis pela transmissão contínua de dados (*streams* ou fluxos). Essas duas abstrações podem ser combinadas formando grafos arbitrários que representem o processamento da aplicação.

Usando esse modelo, criar uma aplicação no *Anthill* é um processo de decomposição do processamento em filtros. Nesse processo, a aplicação é modelada como uma computação no modelo *dataflow* dividida em uma rede de filtros que transformam os dados. Durante a execução, o processo definido para cada filtro é instanciado em diferentes nós do ambiente distribuído. A esses processos dá-se o nome de cópias transparentes ou instâncias de um filtro. Dessa forma, cada estágio do processamento pode ser distribuído por muitos nós de uma máquina paralela e os dados que devem fluir por aquele filtro podem ser particionados pelas cópias transparentes, produzindo o paralelismo de dados desejado.

Nosso modelo de programação, dessa forma, nos permite extrair o máximo de paralelismo das aplicações através das três possibilidades discutidas anteriormente: paralelismo de dados, de tarefas e assincronia. Já que as unidades de processamento são na verdade cópias de estágios de um *pipeline*, pode-se ter um paralelismo de grão fino. Como todo esse processamento pode ocorrer assincronamente, a execução não terá qualquer tipo de retenção (*bottleneck*) devida ao sistema. Para garantir a redução da latência durante o processamento, a granulosidade das tarefas deve ser definida pelo projetista da aplicação.

3.2. Estudos de Caso

Podemos avaliar a aplicabilidade da nossa estratégia de paralelização avaliando a eficiência das classes de aplicações que foram paralelizadas utilizando o ambiente Anthill. Um aspecto importante de ser ressaltado é que a grande virtude desse ambiente é que o paralelismo a ser explorado é limitado apenas pelas dependências de dados inerentes à própria aplicação, as quais podem ser expressas por um grafo acíclico direcionado. Mais ainda, o ambiente permite que essas oportunidades de paralelismo sejam exploradas tanto quanto possível na plataforma computacional provida, uma vez que ela busca utilizar os vários processadores de acordo com as demandas de processamento inerentes ao caminhamento que a aplicação realiza no grafo de dependência de dados.

No contexto do nosso modelo de programação, cada dado está associado a uma tarefa que é responsável por calculá-lo. É interessante notar que a noção de tarefa é variável e depende da granulação do problema e da capacidade de processamento de cada máquina. Sem perda de generalidade, vamos assumir que cada dado a ser calculado pela aplicação está associado a uma tarefa, mas é importante ressaltar que na prática esse dado

pode ser bastante complexo e representado por uma estrutura hierárquica.

Podemos caracterizar esses grafos de dependência de dados, e portanto as aplicações que eles representam, em termos de três critérios: regularidade, dinamismo e monotonicidade. Aplicações regulares são aquelas cujo custo de execução independe da natureza da entrada, em contraste com aplicações irregulares. Há um grande número de aplicações regulares e elas se beneficiam pouco do ambiente Anthill, embora o ambiente possa ser usado de forma transparente para explorar oportunidades de paralelismo de tarefas e de dados. Já no caso de aplicações irregulares, a possibilidade de assincronia provê a flexibilidade necessária à aplicação para que ela seja tão eficiente quanto possível para todas as entradas. Aplicações dinâmicas são aquelas cujo grafo de dependência de dados é conhecido apenas em tempo de execução, o que não acontece com aplicações estáticas. O ambiente Anthill permite uma melhor exploração das oportunidades de paralelismo tanto para aplicações estáticas (desde que elas sejam irregulares) quanto para aplicações dinâmicas. Neste caso, o fato de explorarmos simultaneamente as três oportunidades de paralelismo permite lidar com questões como granulação variável de computação por tarefa e mesmo número variável de oportunidades de paralelismo. Aplicações monotônicas são aquelas onde vértices não são retirados do grafo de dependência de dados, ao contrário de aplicações não-monotônicas. Novamente, ambos os casos se beneficiam do ambiente Anthill. Aplicações monotônicas são acomodadas naturalmente no ambiente, tendo em vista o espaço de armazenamento que pode ser associado a cada identificação de fluxo. No caso de aplicações não-monotônicas, as mudanças de estrutura são absorvidas de forma transparente pelo ambiente, que passa a considerar a nova configuração a partir dos filtros que participam da mudança. Nem todas as configurações de aplicações são plausíveis. Por exemplo, não há aplicações que sejam estáticas e não monotônicas, assim como regulares e dinâmicas.

Demonstramos a generalidade da nossa abordagem realizando a paralelização de várias aplicações. Um sumário dos resultados é apresentado na Tabela 1. Nesta tabela, além do nome da aplicação, indicamos se ela é regular (Reg.), estática (Est.) e monotônica (Mon.). Apresentamos também o melhor resultado que obtivemos em termos de eficiência e o número de processadores utilizados. Todas as execuções utilizaram o ambiente Anthill. A seguir apresentamos uma breve descrição dessas aplicações. O algoritmo para determinação de agrupamentos k-means é um caso degenerado do grafo de dependência de dados, onde o grafo têm apenas um vértice, o que caracteriza aplicações iterativas com sincronização global como barreiras (ou seja, há uma sincronização explícita que incorpora as dependências de dados [11]. A placenta [26] é uma aplicação de processamento digital de imagens, onde devem ser aplicados vários operadores em uma dada ordem para analisar a imagem. Apriori é um algoritmo para geração de regras de associação, onde o grafo de tarefas é conhecido, mas o custo das tarefas não, assim como quanto do grafo é efetivamente calculado [32]. O algoritmo ID3 é um algoritmo para construção de árvores de decisão. O seu grafo de tarefas é determinado em tempo de execução, mas ele não reverte tarefas executadas [11]. O algoritmo Partricluster [1] determina agrupamentos em várias dimensões simultaneamente e possui as mesmas características. O algoritmo Cobweb determina agrupamentos hierárquicos, onde o grafo de tarefas é construído dinamicamente e muda ao longo da execução [10]. O algoritmo Atalaia detecta exceções em bases de dados e possui as mesmas características do CobWeb neste sentido. Podemos notar que em todos os casos obtivemos resultados, se não excelentes, satisfatórios, ates-

Aplicação	Reg.	Est.	Mon	Efic.	Procs
k-Means	S	S	S	92	16
Placenta	S	S	S	90	16
Apriori	N	S	S	95	32
ID3	N	N	S	88	16
Partricluster	N	N	S	65	14
Cobweb	N	N	N	99	8
Atalaia	N	N	N	74	8

Tabela 1. Características e eficiência de diversas paralelizações de algoritmos de mineração de dados

tando a aplicabilidade da nossa estratégia e a diversidade de algoritmos que podem ser paralelizados.

É importante ressaltar que, embora as paralelizações aqui apresentadas sejam de algoritmos tradicionais de mineração de dados, acreditamos que os compromissos na paralelização dos algoritmos de mineração de padrões complexos sejam similares.

4. Discussão

Para que possamos entender as complexas interações sociais que a Internet vem fomentando e permite criar é necessário o desenvolvimento de novas técnicas de mineração de dados que vão permitir a extração automatizada de informações e a realização de análise de volumes de dados cada vez maiores e mais complexos. Neste artigo demonstramos que a construção de tais algoritmos é possível e mostramos como temos resolvido algumas peculiaridades inerentes a essas aplicações. Mais ainda, argumentamos que é fundamental o desenvolvimento de técnicas escaláveis e eficientes para esses cenários e que a obtenção de tais técnicas é através da paralelização. Neste sentido, descrevemos o ambiente Anthill e apresentamos alguns resultados de paralelização de aplicações neste ambiente, as quais demonstram a sua adequação à tarefa.

Os resultados apresentados são na direção de atacar três grupos de desafios de pesquisa em Ciência da Computação: extração automatizada de conhecimento de dados oriundos aplicações Internet, mineração de padrões complexos e escalabilidade e eficiência das técnicas de mineração. A seguir discutimos cada um desses grupos de desafios.

A extração automatizada de conhecimento a partir de dados de aplicações Internet é um desafio importante por várias razões. A primeira delas é a relevância que essas aplicações têm não apenas em termos acadêmicos, mas da sociedade em geral. A segunda razão é que temos testemunhado um aumento de diversidade e de usuários dessas aplicações, trazendo à tona não apenas novas questões em termos da aplicação em si e a análise do seu uso, como da sua escalabilidade. Neste contexto, o projeto e implementação de técnicas e tecnologias que melhorem essas aplicações e permitam analisar o comportamento de usuários representa um desafio contínuo e em constante evolução. Responder a esse desafio transcende a área de Computação, uma vez que é necessário entender o contexto onde essas aplicações estão inseridas e como esse contexto afeta o seu funcionamento e análise. Embora o nosso texto tenha enfocado aplicações

textuais, os compromissos para outras mídias é ainda mais severo, uma vez que o volume em geral é maior e tarefas de coleta e seleção de dados são mais complexas. A nossa visão é que a extração de conhecimento automatizada em várias mídias apresentam desafios similares, de tal forma que as experiências obtidas com textos podem subsidiar a criação de técnicas que se apliquem às outras mídias.

O segundo conjunto de desafios está associado à mineração de padrões complexos, os quais devem considerar três aspectos: qualidade do modelo, natureza dos dados e escopo da mineração. A qualidade do modelo demanda uma reavaliação das técnicas existentes, em particular aquelas que geram modelos globais, os quais podem ser imprecisos pela alta dimensionalidade dos problemas. A natureza dos dados demanda que saibamos lidar com evidências heterogêneas, as quais também afetam a qualidade do modelo. Finalmente, como discutido, o impacto temporal afeta não só a qualidade do modelo como a quantidade de informação provida pelo dado, o que demanda novas técnicas para incorporar tal dimensão tanto em modelos quanto na utilização dos dados, o que não é feito atualmente.

O terceiro desafio é como implementar essa diversidade de técnicas a serem aplicadas nesses vários contextos de uma forma escalável e eficiente, num ambiente computacional que varia de arquiteturas com múltiplos *cores* a grades computacionais. Técnicas de mineração de dados são normalmente difíceis de paralelizar tendo em vista a sua irregularidade e complexidade, o que demanda a exploração das várias dimensões de paralelismo existentes, conforme trabalhos anteriores demonstram. Acreditamos que esse desafio será amplificado pelas novas técnicas de mineração de padrões complexos, mas a natureza da solução será muito semelhante ao que apresentamos.

Em termos de algoritmos de mineração de dados, os trabalhos futuros são na direção de aperfeiçoar e ampliar o escopo dos algoritmos existentes. Os casos de utilização que discutimos são passos iniciais numa grande gama de possibilidades. Como discutido, já temos vários exemplos de algoritmos sobre bases essencialmente estáticas. O projeto de algoritmos evolutivos, que considerem aspectos temporais já está em execução e um novo aspecto é com relação a que esses algoritmos sejam incrementais, evitando o processamento dos mesmos dados a todo momento que os modelos forem retificados.

Atualmente, temos um modelo de programação validado e sólido, no sentido de ter sido e estar sendo utilizado para uma grande gama de aplicações. Entretanto, algumas limitações já são patentes. A primeira é o fato de o ambiente não suportar mudanças na sua configuração em termos de instâncias de filtros e fluxos de forma dinâmica. Tais mudanças são motivadas por uma busca por maior eficiência ou mesmo tolerância a falhas. Em particular, a melhor configuração de uma aplicação em termos de filtros pode se alterar durante a execução de um programa [11] ou mesmo a disponibilidade de processadores pode mudar em conseqüência de uma falha [5]. A segunda limitação é que a implementação de programas paralelos na plataforma ainda é uma tarefa laboriosa, demandando técnicas de paralelização automatizada. O fato do modelo de programação demandar apenas informações sobre dependência de dados facilita esta tarefa, mas a geração automática de aplicações paralelas ainda é um desafio. A terceira é que o modelo de execução é basicamente uni-programado, ou seja, não há como contornar as frações seriais da aplicação. A quarta é que, apesar de já haver sobreposição em termos de funções entre as técnicas paralelizadas, essas semelhanças não podem ser exploradas, pois um

filtro responde a uma e somente uma aplicação.

Todas essas restrições motivam o desenvolvimento de novos modelos de programação e execução de múltiplos programas simultaneamente, maximizando a utilização das máquinas, das funcionalidades e dos dados que já estejam residentes no ambiente de execução. Como trabalhos futuros vamos delinear as bases de um novo modelo de programação que permita a definição de fluxos de trabalho. A diferença fundamental desses fluxos em relação ao modelo atual é que dois outros tipos de informação serão explicitados e explorados para fins de uma paralelização escalável e eficiente: os dados a serem processados e os procedimentos a serem realizados por cada aplicação. Desta forma, seremos capazes de permitir que várias aplicações compartilhem tanto dados quanto procedimentos maximizando a eficiência do ambiente, seja pela redução do volume de comunicação, seja pela reutilização de código e da sua capacidade de execução. Resultados preliminares no contexto de uma aplicação de imagens médicas demonstram esse potencial [26].

Referências

[1] R. Araujo, G. Ferreira, G. Orair, W. Meira Jr., R. Ferreira, D. Guedes, and M. Zaki. The partricluster algorithm for gene expression analysis. *International Journal of Parallel Programming*, 2007.

[2] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison-Wesley, 1999.

[3] I. Bhattacharya and L. Getoor. Iterative record linkage for cleaning and integration. In *Proc. of the 9th ACM SIGMOD workshop on Research issues in data mining and knowledge discovery*, pages 11–18, 2004.

[4] Harold Borko and Myrna Bernick. Automatic document classification. *J. ACM*, 10(2):151–162, 1963.

[5] B. Coutinho. Desempenho e disponibilidade em sistemas distribuídos em larga escala. Master's thesis, DCC – UFMG, Belo Horizonte, Minas Gerais, Brazil, 2005.

[6] N. Dalvi, P. Domingos, Mausam, S. Sanghai, and D. Verma. Adversarial classification. In *Proc. of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 99–108, 2004.

[7] Jochen Darre, Peter Gerstl, and Roland Seiffert. Text mining: finding nuggets in mountains of textual data. In *KDD '99: Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 398–401, New York, NY, USA, 1999. ACM Press.

[8] Ismail Fahmi. Examining learning algorithms for text classification in digital libraries. In *Master's thesis*, Groninge, Netherland, 2004.

[9] Tom Fawcett. 'in vivo' spam filtering: A challenge problem for data mining. *KDD Explorations*, 5(2), December 2003. Disponível em: `http://www.hpl.hp.com/personal/Tom_Fawcett/papers/spam-KDDexp.pdf`.

[10] G. Ferreira, R. Araújo, G. Orair, L. Gonçalves, D. Guedes, R. Ferreira, W. Meira Jr., and V. Furtado. Paralelização eficiente de um algoritmo de agrupamento hierárquico.

In *Anais do Workshop sobre Algoritmos de Mineração de Dados*, Uberlândia, MG, 2005.

[11] R. Ferreira, W. Meira Jr., D. Guedes, L. Drummond, B. Coutinho, G. Teodoro, T. Tavares, R. Araújo, and G. Ferreira. Anthill: A scalable run-time environment for data mining applications. In *Proc. of the 17th International Symposium on Computer Architecture and High Performance Computing*, Rio de Janeiro, RJ, 2005.

[12] Jerome H. Friedman, Ron Kohavi, and Yeogirl Yun. Lazy decision trees. In Howard Shrobe and Ted Senator, editors, *Proceedings of the Thirteenth National Conference on Artificial Intelligence and the Eighth Innovative Applications of Artificial Intelligence Conference*, pages 717–724, Menlo Park, California, 1996. AAAI Press.

[13] D. Guedes, W. Meira Jr., and R. Ferreira. Anteater: A service oriented architecture for high-performance data mining. *IEEE Internet Computing*, 10(4):36–43, 2006.

[14] T. Joachims. Making large-scale svm learning practical. In *Scholkopf B., Burges C.J.C., and Smola A.J. (Eds.), Advances in Kernel Methods-Support Vector Learning*, pages 169–184, Cambridge, MA, 1999. MIT Press.

[15] Matthew B. Koll. Information retrieval theory and design based on a model of the user's concept relations. In *SIGIR '80: Proceedings of the 3rd annual ACM conference on Research and development in information retrieval*, pages 77–93, Kent, UK, UK, 1981. Butterworth & Co.

[16] R. Kosala and H. Blockeel. Web mining research: A survey. *SIGKDD: SIGKDD Explorations: Newsletter of the Special Interest Group (SIG) on Knowledge Discovery and Data Mining, ACM*, 2, 2000.

[17] J. Kubica, A. Moore, and J. Schneider. Tractable group detection on large link data sets. In *Proc. of the Third IEEE International Conference on Data Mining*, 2003.

[18] Wai Lam and Chao Yang Ho. Using a generalized instance set for automatic text categorization. In *SIGIR '98: Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 81–89, New York, NY, USA, 1998. ACM Press.

[19] Q. Lu and L. Getoor. Link-based classification. In *Proc. of the 2003 International Conference on Machine Learning*, pages 496–503, 2003.

[20] A McCallum and K Nigam. A comparison of event models for naive bayes text classification. In *Workshop on Learning for Text Categorization*. AAAI, 1998.

[21] A. Pereira, F. Mourao, P. Goes, and W. Meira Jr. Reactivity in online auctions. In *Proc. of Workshop Reactivity on the Web at the International Conference on Extending Database Technology (EDBT'2006)*, volume 3896, Munich, 2006. Springer – Lecture Notes in Computer Science.

[22] A. Pereira, L. Silva, and W. Meira Jr. Evaluating the impact of reactive workloads on the performance of web applications. In *Proc. of the IEEE International Performance Computing and Communications Conference (IPCCC), 2006*, volume 1, pages 425–432, Phoenix, 2006.

[23] A. Pereira, L. Silva, W. Meira Jr., and W. Santos. Assessing the impact of reactive workloads on the performance of web applications. In *Proc. of the IEEE International*

Symposium on Performance Analysis of Systems and Software (ISPASS), 2006, volume 1, pages 425–432, Austin, TX, 2006.

[24] B. Possas, N. Ziviani, B. Ribeiro-Neto, and W. Meira Jr. The set-based model for information retrieval. *ACM Transactions on Information Systems*, 23(4):397–429, 2005.

[25] J. P. Scott. *Social Network Analysis: A handbook*. Sage publications, 2005.

[26] G. Teodoro, T. Tavares, R. Ferreira, T. Kurc, W. Meira Jr., D. Guedes, and J. Saltz. Runtime support for efficient execution of scientific workflows on distributed environments. In *Proc. of the XVIII International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD), 2006*, Ouro Preto, MG, 2006. IEEE.

[27] Loren Terveen, Will Hill, and Brian Amento. Constructing, organizing, and visualizing collections of topically related web resources. *ACM Trans. Comput.-Hum. Interact.*, 6(1):67–94, 1999.

[28] A. Veloso, M. Cristo, W. Meira Jr., M. Goncalves, and M. Zaki. Multi-evidence, multi-criteria, lazy associative classification. In *Proc. of the ACM Fifteenth Conference on Information and Knowledge Management - CIKM 2006*, Arlington, VA, EUA, 2006.

[29] A. Veloso and W. Meira Jr. Eager, lazy and hybrid algorithms for multi-criteria associative classification. In *Anais do Workshop sobre Algoritmos de Mineração de Dados (WAMD) 2005*, pages 17–25, Uberlândia,MG, 2005.

[30] A. Veloso and W. Meira Jr. Rule generation and rule selection techniques for cost-sensitive associative classification. In *Anais do Simpósio Brasileiro de Bancos de Dados, 2005*, pages 295–309, Uberlândia,MG, 2005.

[31] A. Veloso and W. Meira Jr. Lazy associative classification for content-based spam detection. In *Proc. of the Fourth Latin American Web Congress - LA-WEB 2006*, Cholula, Mexico, 2006.

[32] A. Veloso, W. Meira Jr., R. Ferreira, D. Guedes, and S. Parthasarathy. Asynchronous and anticipatory filter-stream based parallel algorithm for frequent itemset mining. In *Proceedings of the 8th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD)*, pages 422–433, Pisa, Italy, 2004.

[33] A. Veloso, W. Meira Jr., T. Macambira, D. Guedes, and H. Almeida. Automatic moderation of comments in a large on-line journalistic environment. In *Proc. of the International Conference on Weblogs and Social Media*, Boulder, CO, USA, March 2007.

[34] A. Veloso, W. Meira Jr., and M. Zaki. Lazy associative classification. In *Proc. of the International Conference on Data Mining - ICDM 2006*, pages 645–654, 2006.

[35] Adriano Veloso, Wagner Meira Jr., Marco Cristo, Marcos Gonçalves, and Mohammed J. Zaki. Multievidence, multicriteria, lazy associative document classification. In *Conference on Information and Knowledge Management (CIKM)*. ACM, 2006.

[36] T. Washio and H. Motoda. State of the art of graph-based data mining. *ACM SIGKDD Explorations Newsletter*, 5(1):59–68, July 2003.