Blocagem Adaptativa e Flexível para o Pareamento Aproximado de Registros

Luiz Osvaldo Evangelista[1], Eli Cortez[1], Altigran S. da Silva[1], Wagner Meira Jr.[2]

[1]Departamento de Ciência da Computação
Universidade Federal do Amazonas (UFAM)
Manaus – AM – Brasil

luizrevangelista@gmail.com {eccv,alti}@dcc.ufam.edu.br

[2]Departamento de Ciência da Computação
Universidade Federal de Minas Gerais (UFMG)
Belo Horizonte – MG – Brasil

meira@dcc.ufmg.br

Abstract. *In data integration tasks, records from a single dataset or from different sources must be often compared to identify records that represent the same real world entity. The cost of this search process for finding duplicate records grows quadratically as the number of records available in the data sources increases and, for this reason, direct approaches, as comparing all record pairs, must be avoided. In this context, blocking methods that are based on machine learning processes are used to find the best blocking function, based on the combination of low cost rules, which define how to perform the record blocking. This work presents a new blocking method based on machine learning. Different from other methods, this new approach is based on genetic programming, allowing the use of more flexible rules and a larger number of such rules for defining blocking functions, leading to a more effective process of identification of duplicate records. Experimental results with real and synthetic data show that the correctness of the genetic programming method may be over 95% when detecting duplicate records in an efficient manner.*

Resumo. *Em tarefas de integração de dados, registros de mesma fonte ou de fontes diferentes precisam ser frequentemente comparados para identificar pares de registros que correspondam à mesma entidade no mundo real. O custo desses processos de busca por registros duplicados cresce quadraticamente com o aumento do tamanho das fontes de dados e por isso as abordagens diretas, analisando todos os pares de registros, devem ser evitadas. Nesse contexto, métodos de blocagem baseados em Aprendizagem de Máquina têm sido usados para encontrar a melhor função de blocagem, sendo essas funções definidas por combinações de regras de baixo custo de processamento que determinam como os registros devem ser agrupados. Esta trabalho apresenta um novo método de blocagem baseado em aprendizagem de máquina. Diferente dos demais métodos, essa nova abordagem é baseada em programação genética, permitindo o uso de regras mais flexíveis e um maior número de regras para a definição de funções de blocagem, aumentando também a eficácia na identificação de registros duplicados. Resultados de experimentos com dados*

reais e sintéticos mostram que percentuais de acertos acima de 95% podem ser conseguidos na detecção de pares duplicados de registros de maneira eficiente.

1. Introdução

Um dos principais problemas encontrados em integração de dados é o de identificar, nos arquivos a serem integrados, registros que correspondam à mesma entidade no mundo real. Nesse contexto, uma entidade pode ser uma empresa, um indivíduo ou qualquer outro elemento encontrado no mundo real e com significado bem definido [Winkler 2006]. Depois de analisados e comparados entre si, os registros identificados podem ser organizados de maneira a formar pares de registros que são considerados duplicados. Esse processo de comparação e identificação de registros replicados é conhecido como *Record Linkage* (RL) ou *Pareamento de Registros*.

Através do pareamento podem ser vinculados registros obtidos de fontes de dados distintas ou podem ser identificadas duplicatas em uma única fonte [Winkler 2006]. O objetivo é o mesmo em ambos os casos, permitindo que os registros de arquivos diferentes sejam associados ou criando condições para que registros duplicados de mesma origem sejam identificados e removidos, melhorando a qualidade dos dados e facilitando o acesso à informação. Um exemplo de associação entre registros duplicados é apresentado na Figura 1.

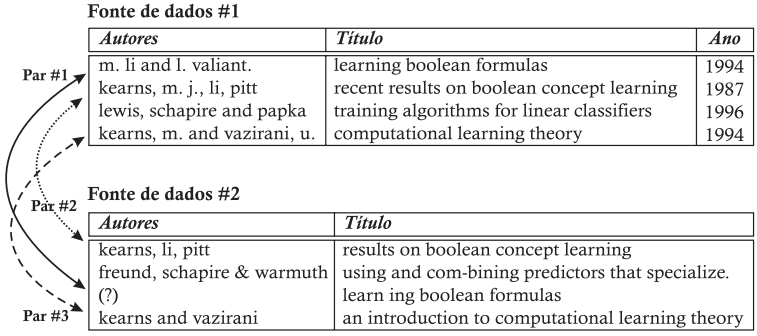

Figura 1. Pareamento de registros usando casamento aproximado.

Como no exemplo ilustrado, pode ser observado na prática que associar registros que representam uma mesma entidade não é uma tarefa trivial, visto que os registros normalmente fornecidos para esse tipo de operação não apresentam identificadores únicos válidos e por isso são usados valores de atributos para determinar a duplicidade em meio a esses dados. Esse é o caso dos atributos *Autores* e *Título* apresentados na Figura 1.

De acordo com o exemplo, pode-se notar que geralmente os valores de atributos não apresentam um padrão de representação (ex. abreviações, pontuação, etc.), dificultando a identificação de pares de registros de mesma entidade por meio de casamento exato entre as *strings* obtidas a partir de cada registro. Por esse motivo, utilizam-se técnicas específicas para que a identificação de pares seja feita de forma aproximada, como, por exemplo, técnicas baseadas em métricas de similaridade de texto, considerando termos semelhantes ao invés do casamento exato de termos. Devido ao uso dessas técnicas, a tarefa de casamento de registros é também chamada de junção *aproximada* de registros.

Outros problemas podem ocorrer, mesmo quando essas métricas de similaridade são usadas. Por exemplo, os arquivos que precisam ser processados, freqüentemente po-

dem conter grandes quantidades de registros e o número de pares formados com esses registros pode demandar muito tempo de processamento [Michelson and Knoblock 2006]. Como exemplo prático, considere o caso em que dois arquivos de dados são compostos por 10 mil registros, e o número de pares formados com os registros desses dois arquivos é de 100 milhões. Se cada um dos pares for avaliado em 0.01 segundo, analisar todos os pares demandaria aproximadamente 11 dias, tempo de processamento considerado grande para tratar os 10 mil registros encontrados em cada conjunto de dados.

Em [Winkler 1994], é apresentada uma abordagem baseada em um algoritmo aproximado para diminuir o número de pares candidatos. Por essa abordagem, as associações entre registros ocorrem na forma *um-para-um*, onde cada registro em um primeiro conjunto é vinculado ao registro com maior grau de similaridade encontrado em um segundo conjunto. O algoritmo que faz essas associações pode não apresentar bons resultados em todos os casos, sendo mais indicado nas situações em que cada conjunto de registros apresenta um número pequeno de duplicatas.

Para produzir bons resultados, independente do número de duplicatas em cada conjunto de dados, métodos de *Blocagem* podem ser usados, servindo como estratégia rápida para agrupar registros, de acordo com algum critério de baixo custo de processamento em um primeiro momento, para que depois sejam identificados os pares de registros com maiores probabilidades de corresponder à mesma entidade.

Depois de realizado o processo de blocagem, os blocos de registros ou os pares identificados como sendo de mesma entidade são usados para facilitar alguma operação de junção aproximada ou deduplicação de registros. Como uma conseqüência da blocagem, é esperado que, ao invés de uma quantidade muito grande de pares de registros, sejam considerados apenas os candidatos mais prováveis para que as operações de pareamento sejam realizadas em menos tempo.

Em geral, os processos de blocagem podem ser usados como fase preparatória para processos de pareamento. Neste sentido, os métodos de blocagem podem ser vistos também como técnicas de poda de informação, descartando os pares que apresentam as menores probabilidades de corresponder à mesma entidade.

Apresentamos neste artigo um novo método de blocagem, denominado *BGP* (Blocagem baseada em Programação Genética), baseado na aprendizagem de expressões de blocagem na forma normal disjuntiva. O problema em questão é encontrar o melhor esquema de blocagem, dado um conjunto de pares de registros para análise em um processo de aprendizagem de máquina e um conjunto de regras para verificações feitas com esses pares de registros. Vale ressaltar que este é o mesmo cenário tratado por outros métodos de aprendizagem de máquina.

Na elaboração do método *BGP*, foi usada a técnica de programação genética (PG) [Koza 1998], que é capaz de verificar e combinar números possivelmente maiores de regras do que as quantidades experimentadas em outros trabalhos na literatura [Bilenko et al. 2006, Michelson and Knoblock 2006]. Regras mais complexas também podem ser usadas nesse caso, aumentando as chances de que resultados melhores sejam alcançados. Sem a restrição para o número e a complexidade de regras, várias possibilidades podem ser experimentadas, como por exemplo o uso de regras baseadas em parâmetros.

O uso de parâmetros pode ser considerado um terceiro aspecto para a composição de esquemas de blocagem. Enquanto métodos anteriores combinam apenas regras e atributos para definir os esquemas de blocagem, o *BGP* escolhe também os melhores valores de parâmetros que podem ser usados em cada regra, fazendo com que os processos de aprendizagem de máquina ocorram com maior flexibilidade. Como resultado do uso desse terceiro elemento, a probabilidade de produzir esquemas de blocagem mais adaptados aos dados é maior, aumentando as chances de que os resultados de blocagem também sejam os melhores.

Usando as vantagens do *BGP*, os processos de agrupamento de registros podem ser realizados com maior flexibilidade, usando um número maior e mais complexo de regras de blocagem, resultando em esquemas de blocagem que permitem a identificação de um número maior de pares duplicados de registros, ao mesmo tempo em que o custo de execução do algoritmo é mantido em um limite aceitável.

Realizamos experimentos para avaliar a qualidade e a escalabilidade do nosso método de blocagem, usando as coleções *Cora*, *CiteSeer* e *Evolbase*. Dessas, as coleções *Cora* e *CiteSeer* são constituídas de dados reais e a coleção *Evolbase*, de dados sintéticos. Essas coleções foram usadas como base em estudos anteriores, sendo a coleção *Cora* usada em [Bilenko et al. 2006], para avaliar o método *DNF Blocking*, e as coleções *CiteSeer* e *Evolbase*, em [Sarawagi and Bhamidipaty 2002] e [de Carvalho et al. 2006]. Além disso, também apresentamos experimentos comparativos entre o método aqui proposto, *BGP*, e os métodos *DNF Blocking* [Bilenko et al. 2006] e o método *BSL Blocking* [Michelson and Knoblock 2006].

O restante do artigo encontra-se organizado da seguinte forma. Na Seção 2 revisamos os métodos de blocagem existentes na literatura corrente e as diferenças entre as abordagens usadas nesses métodos. Na Seção 3 o método *BGP* é apresentado como um método de blocagem alternativo, descrevendo como foi elaborado usando os recursos da programação genética. A Seção 4 apresenta os resultados dos experimentos realizados com o método proposto, comparando-o com os métodos que foram utilizados como linha de base. Por fim, a Seção 5 apresentadas as conclusões deste trabalho, assim como sugestões de trabalhos futuros.

2. Trabalhos Relacionados

Vários estudos sobre métodos de blocagem podem ser encontrados na literatura atual. As primeiras propostas foram baseadas no uso de métricas de similaridade e por isso podem falhar em várias situações, por exemplo quando os arquivos apresentam quantidades grandes de duplicatas [Bhattacharya and Getoor 2004]. Genericamente, os métodos de blocagem poder ser classificados em dois grupos de acordo com a abordagem usada para organizar os registros em blocos: (1) *métodos estáticos* e (2) *métodos dinâmicos*.

Nos métodos estáticos de blocagem, o processo de agrupamento dos registros não leva em consideração as características encontradas nos dados, e são realizados da mesma forma em todas as situações. Exemplos de métodos com essa característica são *Canopy Blocking* [McCallum et al. 2000] e *Soft TF-IDF* [Cohen et al. 2003, Cohen et al.].

A idéia do método *Canopy Blocking* é agrupar registros de forma eficiente em um processo de duas etapas, onde a primeira etapa é realizada com baixo custo de processamento e a segunda é usada com o objetivo de refinar os resultados da primeira etapa.

Deve ser notado, contudo, que essa abordagem pode não funcionar em todos os casos. Por exemplo, entidades diferentes podem apresentar registros semelhantes, fazendo com que sejam associadas aos mesmos blocos. Como resultado de falhas de blocagem como essa, um número maior de pares falsos de registros pode ser formado a partir dos blocos gerados e, conseqüentemente, o número de pares candidatos retornado é aumentado desnecessariamente.

Como alternativa para o método *Canopy Blocking*, métodos baseados em abordagens diferentes para o cálculo de similaridade podem ser utilizados, capturando, por outros meios, as noções de duplicidade entre registros. Outra abordagem para a verificação da similaridade entre registros pode ser encontrada no método *Soft TF-IDF*. Tal método foi proposto inicialmente como uma métrica de similaridade de texto, permitindo descobrir o grau de similaridade entre registros diferentes, considerando um atributo por vez.

Métodos mais recentes de blocagem são baseados em processos de Aprendizagem de Máquina (*Machine Learning*) para determinar a melhor função de blocagem para o agrupamento ou comparação de registros e por isso são conhecidos como dinâmicos e adaptativos. Esses métodos são diferentes dos métodos estáticos, uma vez que os estáticos são baseados em regras de agrupamento predefinidos e não mudam de acordo com as características encontradas nos registros. Dois exemplos de métodos de blocagem dinâmicos e adaptativos são *DNF Blocking* [Bilenko et al. 2006] e *BSL Blocking* [Michelson and Knoblock 2006].

As instâncias de treinamento para aprendizagem de máquina realizada com o método *DNF Blocking* são pares de registros classificados como verdadeiros, se apresentam registros de mesma entidade, ou falsos, se os registros são de entidades diferentes. Essas instâncias de treinamento são usadas em análises para a escolha de regras que produzam os melhores agrupamentos de registros na blocagem. Essas regras são selecionadas e combinadas para formar expressões na forma normal disjuntiva (FND) chamadas de *esquemas de blocagem*. Essas expressões recebem esse nome, uma vez que definem a forma como os registros são agrupados.

O método *DNF Blocking* pode apresentar resultados melhores nos casos em que os métodos estáticos falham, visto que as abordagens dinâmicas e adaptativas permitem capturar a noção de duplicidade entre registros sem necessariamente considerar a similaridade entre os valores de seus atributos. Porém, como dito anteriormente, um aumento do número dessas regras pode significar um aumento também do tempo necessário para o processo de aprendizagem de máquina, dificultando o uso do método *DNF Blocking*. Como alternativa para reduzir o tempo de processamento, pode ser usado o método *BSL Blocking*.

O método *BSL Blocking* é semelhante ao *DNF Blocking* no que se refere ao uso de regras de blocagem, apresentando maiores diferenças na forma como as amostras de treino são usadas e no método de combinação dos predicados para produzir os esquemas de blocagem nos processos de aprendizagem de máquina.

Na prática, pode ser observado que o método *BSL Blocking* é normalmente usado com amostras contendo números pequenos de pares de registros de treino, por considerar apenas os pares verdadeiros das amostras de treino. Por esse motivo pode apresentar tempos menores de execução, comparados aos tempos de execução do método *DNF Blocking*.

Essa vantagem pode ser usada nos casos em que o tempo curto de processamento for um requisito crítico para o agrupamento dos registros.

3. O Método BGP

Nesta seção descrevemos o método *BGP* (Blocagem baseada em Programação Genética), que utiliza programação genética (PG) como base para resolver o problema de blocagem adaptativa. O método *BGP*, assim como os métodos *DNF Blocking* e *BSL Blocking*, são baseados em esquemas de blocagem formados a partir de predicados *booleanos* para a identificação dos pares de registros que correspondem à mesma entidade no mundo real. Ou seja, o método *BGP* também requer amostras de dados para descobrir bons esquemas de blocagem, usando aprendizagem de máquina para atingir esse objetivo.

Contudo, o método *BGP* diferencia-se dos métodos baseados em aprendizagem de máquina, no que diz respeito à forma como os predicados de blocagem são combinados no processo de procura pelo melhor esquema de blocagem. Para atingir esse objetivo, os métodos de blocagem anteriores foram desenvolvidos usando algoritmos iterativos que analisam combinações de predicados de blocagem, os quais são verificados em uma determinada *seqüência*, de acordo com os respectivos algoritmos.

Para efeito de discussão, as regras usadas nos métodos anteriores foram chamadas de regras *standard* e as regras da segunda versão do método *BGP*, chamadas de *regras baseadas em parâmetros*. Logo, a versão do nosso método que utiliza regras *standard* é designada *BGP-SR* e a versão de regras baseadas em parâmetros, designada *BGP-PR*.

3.1. Uso de Programação Genética em Blocagem Adaptativa

A abordagem de programação genética pode ser usada para resolver problemas de blocagem adaptativa com a idéia de que os esquemas de blocagem são os programas de computador que passam pelo processo de evolução. Essa idéia é desenvolvida a seguir.

3.1.1. Representação de Esquemas de Blocagem

Em nossa abordagem, os esquemas de blocagem são representados como árvores ao longo do processo de evolução para que sejam modificados com o uso de operadores genéticos. Originalmente, entretanto, os esquemas de blocagem são modelados como seqüências de predicados [1] e operadores *booleanos* intercalados. Como exemplo dessa organização, pode ser apresentado o esquema baseado na expressão:

$$\frac{(\{\text{endereço, bigramasEmComum}\}\ \&\&\ \{\text{veículo, termosEmComum}\}\ \&\&\ \{\text{editora, primeiros3CaracteresCoincidentes}\})}{(\{\text{título, termosEmComum}\})}$$

Por essa expressão, dois registros são considerados referentes à mesma entidade, se apresentam algum termo em comum em valores do atributo *título* ou, caso essa possibilidade não ocorra, se *bigramas* em comum forem encontrados em valores do atributo *endereço*, ao mesmo tempo em que algum termo em comum for encontrado usando o atributo *veículo* e o atributo *editora* apresentar valores com os primeiros três caracteres coincidentes.

[1] No método *BGP*, os predicados são semelhantes aos obtidos com os métodos *DNF Blocking* e *BSL Blocking*, sendo formados por combinação de atributos e regras de afinidade.

O esquema de blocagem do exemplo anterior pode ser visto na forma de árvore, como apresentado na Figura 2.

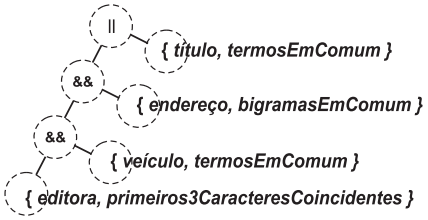

Figura 2. Representação de esquema de blocagem.

Na Figura 2, cada nodo terminal corresponde a um predicado de blocagem e os demais nodos são definidos pelo operador *booleano ou* (representado por "——") ou pelo operador *e* (identificado por "&&"). Cada aresta, representa o vínculo entre predicados e operadores *booleanos*, definindo as associações entre elementos desses tipos para a representação dos esquemas de blocagem na forma de árvores.

Com a representação na forma de árvore, maior flexibilidade pode ser conseguida para a extração de segmentos das expressões de esquemas de blocagem, facilitando o uso dos operadores genéticos, uma vez que esses são baseados em partes de definições de esquemas de blocagem. Os esquemas de blocagem na forma normal disjuntiva são preferidos por apresentarem chances maiores de produzir bons agrupamentos de registros. A forma normal disjuntiva facilita o entendimento de que esquemas de blocagem são uma coleção de conjunções de predicados. Para identificar os esquemas com maiores chances de produzir bons resultados, pode ser usada uma função de *fitness* eficiente. Tal função é descrita a seguir.

3.1.2. Função de *Fitness*

Os esquemas de blocagem são considerados bons quando ajudam na identificação eficiente de todos ou de grande parte dos pares de registros duplicados. Para que essa qualidade seja medida, permitindo a comparação entre diversos esquemas e a seleção da melhor opção entre eles durante um processo de aprendizagem de máquina, por exemplo, algum critério baseado na cobertura de pares de registros deve ser usado, como ocorre em outros métodos adaptativos de blocagem. Critérios comuns são a *cobertura de pares verdadeiros* (*PC*) e a *redução do número de pares candidatos* (*RR*).

No método *BGP*, para medir a qualidade dos esquemas de blocagem, os critérios *PC* e *RR* são usados de forma combinada, gerando um único valor para ser usado como grau de *fitness*. Usando essa combinação de valores, o melhor esquema de blocagem é então o que apresenta os melhores resultados para ambos os critérios de qualidade. Essa combinação pode é feita com uso da medida harmônica *F*.

Para uso da medida harmônica *F* em sistemas de blocagem, os critérios *precisão* e *revocação*, originários da medida *F*, são substituídos pelos critérios *PC* e *RR*. A função de *fitness* pode então ser descrita como baseada em um conjunto $R = \{r_1, ...r_n\}$ de registros, de onde são obtidos os registros de treino $R_T \subset R$. O conjunto de dados de treino é definido como $T = (P, L)$, onde $P = \{P'_1, ...P'_m\}$, $P'_i = \{(r_j, r_k) \in R_T \times R_T | j \neq k\}$, correspondem aos pares de treino, e $L = \{l_1, ...l_m\}$, $l_i \in \{0, 1\}$, aos respectivos rótulos de cada $P'_i \in P$. Nessa definição, os pares $P_i \in P$ são ditos verdadeiros se $l_i = 1$ e

falsos, se $l_i = 0$. Usando a medida harmônica *F*, o número de pares verdadeiros e falsos de registros corretamente identificados são usados no cálculo de *fitness*.

Além disso, vale ressaltar que esquemas de blocagem com números maiores de conjunções de regras de afinidade podem cobrir números maiores de pares verdadeiros de registros. Logo, essa idéia pode ser incorporada à função de fitness, definindo a função apresentada na Equação 1.

$$f^*_{FIT} = f_{FIT} + \left[\frac{C}{100}\right] \quad (1)$$

onde C representa o número de conjunções encontradas no esquema de blocagem avaliado.

A função de *fitness* representada na Equação 1 foi usada como forma de avaliação dos esquemas blocagem durante as etapas de aprendizagem de máquina realizadas nos experimentos reportados neste artigo.

3.1.3. Algoritmo de PG e Parâmetros de Configuração

Adaptado para resolver o problema de blocagem, o algoritmo genérico de PG descrito em [Koza 1998] pode ser visto como a seqüência de passos a seguir:

- Gerar aleatoriamente um conjunto inicial de esquemas de blocagem (indivíduos);
- Avaliar os esquemas, associando um valor calculado pela função de *Fitness* a cada um deles;
- Criar um novo conjunto de esquemas (nova geração):
  - Copiando os melhores esquemas para o novo conjunto;
  - Criando novos esquemas por meio de mutação;
  - Criando novos esquemas por meio de *crossover* (reprodução);
  - Avaliando os novos esquemas, associando-os aos valores de (*Fitness*) correspondentes.
- O melhor esquema após um número predeterminado de gerações é indicado como solução de blocagem adaptativa.

Comparando o algoritmo adaptado ao algoritmo genérico de programação genética, pode ser percebido que a adaptação ocorre basicamente incorporando o conceito de esquema de blocagem à estrutura do algoritmo genérico, substituindo o conceito geral de indivíduo pelo conceito de esquema de blocagem. No algoritmo, também podem ser percebidos alguns parâmetros implícitos à estrutura do algoritmo genérico.

Entre esses parâmetros, podem ser considerados principais os identificados por (1) número de gerações, (2) profundidade máxima da árvore de representação dos indivíduos, (3) número de indivíduos em cada geração, (4) número de melhores indivíduos copiados para geração posterior, (5) probabilidade de ocorrência de mutação e (6) profundidade máxima em segmentos de mutação.

3.2. *BGP-SR* e *BGP-PR*

O método *BGP* foi implementado em duas versões, sendo a *BGP-SR* uma delas, baseada em regras *standard*, como as verificadas nos métodos *DNF Blocking* e *BSL Blocking*. A outra versão, identificada como *BGP-PR*, foi definida e experimentada com o uso regras baseadas em parâmetros.

Entre as duas versões experimentadas, a *BGP-SR* pode ser vista como a mais próxima dos métodos *DNF Blocking* e *BSL Blocking*, uma vez que o *BGP-SR* e esses métodos foram verificados com o mesmo conjunto de regras de blocagem. Nesse sentido, a versão *BGP-PR* apresenta diferenças, visto que foi definida com base em regras baseadas em parâmetros, sendo esse conjunto maior do que o conjunto usado nos demais métodos.

Considerando que os conjuntos de regras podem ser escolhidos para que os processos de blocagem do método *BGP* sejam iniciados, o processo de blocagem desde a seleção de regras até a obtenção dos pares candidatos de registros pode ser visto como apresentado na Figura 3.

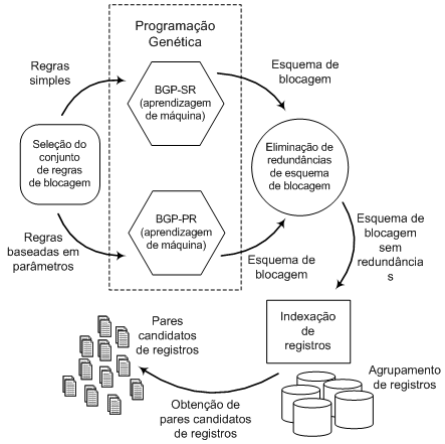

Figura 3. Blocagem usando os métodos *BGP-SR* e *BGP-PR*.

No esquema da Figura 3 é assumido que a escolha do conjunto de regras de blocagem define a versão do método *BGP* que deve ser usada para o agrupamento de registros em cada caso. Em seguida, o agrupamento pode ser visto como um processo de duas fases. Em um primeiro momento, o algoritmo de programação genética é usado para produzir o esquema de blocagem que deve ser usado para o agrupamento de registros na segunda fase. Com esse propósito, são usadas amostras de treino apresentando pares verdadeiros e falsos de registros para aprendizagem de máquina. Na segunda fase, o processo de indexação dos registros é realizado com uma passagem única pelos registros que devem ser agrupados, associando cada registro aos respectivos blocos.

3.3. Blocagem usando regras *standard* - *BGP-SR*

Um elemento importante que pode ser verificado nos métodos de blocagem adaptativa é o algoritmo usado para combinar as regras que são usadas para definir os esquemas de blocagem em processos de aprendizagem de máquina. As regras *standard* são: (1) Igualdade perfeita, (2) Termo em comum, (3) Bigrama em comum, (4) Tetragrama em comum, (5) Hexagrama em comum, (6) Primeiro caractere coincidente, (7) Três primeiros caracteres coincidentes e (8) Cinco primeiros caracteres coincidentes.

Aproveitando as vantagens do processo aleatório de geração de esquemas de blocagem, um maior número de regras pode ser usado na programação genética, aumentando a diversidade de formas disponíveis para a análise de pares de registros, permitindo gerar

esquemas de blocagem mais adaptados aos conjuntos de dados processados. Contudo, o uso de conjuntos de regras maiores pode requerer um esforço maior de preparação para a blocagem de registros, uma vez que essas regras são usadas para a cobrir casos específicos de duplicação de registros, quando selecionadas e combinadas para compor os esquemas de blocagem, e essas regras são definidas uma por vez.

Usando regras baseadas em parâmetros, que podem servir para aumentar o a variedade de regras de blocagem que podem ser consideradas no processo de aprendizagem de máquina, os resultados de experimentos podem ser verificados em condições diferentes, permitindo ter uma idéia do quão melhor pode ser a abordagem de programação genética, considerando quantidades de regras possivelmente maiores do que as quantidades usadas em outras abordagens. As regras baseadas em parâmetros usadas nos experimentos da versão *BGP-PR* são descritas na Seção 3.4.

3.4. Blocagem usando regras baseadas em parâmetros - *BGP-PR*

Como forma de conseguir uma grande variedade de predicados, as regras podem ser baseadas em parâmetros com o objetivo de permitir variações na maneira como os registros podem ser comparados, aumentando o número de casos de duplicação que podem ser previstos quando bons esquemas de blocagem são procurados em etapas de aprendizagem de máquina.

Em experimentos realizados com o método *BGP-PR*, as regras baseadas em parâmetros usadas foram definidas com base nas regras *termos em comum* e *n-gramas em comum*, que são semelhantes às regras *termos em comum* e *bigramas/trigramas/hexagramas em comum* do conjunto de regras *standard* usado nos métodos de blocagem *DNF Blocking* e *BSL Blocking*. As regras baseadas em parâmetros consideradas para a definição do método *BGP-RP* são apresentadas na Tabela 1.

Regras	Parâmetros
1. Termos em comum	Número de termos consecutivos em comum
2. N-gramas em comum	Tamanho de substrings em comum

Tabela 1. Regras baseadas em parâmetros para a análise de pares de registros.

A regra *termos em comum* pode ser usada para identificar pares de registros de mesma entidade que apresentam um determinado número de termos consecutivos em comum. Nesse caso, o parâmetro definido corresponde ao número mínimo de termos que pode ser usado para identificar os casos de registros duplicados, como pode ser verificado no par de registros:

registro#1: *title*:Models of Machines and Computation for Mapping in Multicomputers
e
registro#2: *title*:Models of Machines for Mapping in Multicomputers,

onde são encontrados os termos *Models*, *of* e *Machines* em comum e nessa ordem. Além desses três termos consecutivos, também são encontrados outros quatro termos consecutivos em comum, sendo esses os termos *for*, *Mapping*, *in* e *Multicomputers*. Para cobrir esses dois casos, a regra *termos em comum* pode ser usada com os valores *três* e

quatro para o parâmetro definido. Usando esses valores de parâmetros, os predicados de blocagem definidos com base em um atributo *title* podem ser representados como {*title, termosEmComum-3*} e {*title, termosEmComum-4*}, respectivamente.

Usando uma variedade maior de regras, é esperado que os processos de blocagem ocorram com maior flexibilidade, definindo blocos com maiores probabilidades de apresentar registros duplicados, uma vez que podem ser encontrados esquemas de blocagem mais adaptados aos dados processados nessas condições.

Os resultados de experimentos usando as regras *standard* e baseadas em parâmetros podem ser vistos na Seção 4, onde os métodos *DNF Blocking*, *BSL Blocking* e *BGP* são avaliados e comparados com base na quantidade de pares de registros de mesma entidade identificados de forma correta e também com relação ao número de pares candidatos retornados como resultado dos processos de blocagem realizados.

4. Experimentos

Realizamos os experimentos para avaliar a qualidade e a escalabilidade dos métodos de blocagem estudados, usando as coleções *Cora*, *CiteSeer* e *Evolbase*. Dessas, as coleções *Cora* e *CiteSeer* são constituídas de dados reais e a coleção *Evolbase*, de dados sintéticos. Essas coleções foram usadas como base em estudos anteriores, sendo a coleção *Cora* usada em [Bilenko et al. 2006], para experimentar o método *DNF Blocking*, e as coleções *CiteSeer* e *Evolbase*, em [Sarawagi and Bhamidipaty 2002] e [de Carvalho et al. 2006].

Nos experimentos de qualidade, procuramos verificar as situações em que os métodos estudados apresentaram bons índices de detecção de pares de registros de mesma entidade. Usando técnicas de blocagem, os pares detectados como sendo de registros duplicados são retornados como um conjunto de pares candidatos.

Configuração dos Experimentos

O critério que mede a cobertura de pares de registros duplicados (*PC*) e o de redução do número de pares candidato (*RR*) podem ser calculados como definido asseguir:

$$PC = \frac{||V_C||}{||V||}, \quad RR = 1 - \left[\frac{||C||}{||T||}\right], \quad CAM = \frac{NCP}{NP} \tag{2}$$

onde $||V_C||$ corresponde ao número de pares identificados corretamente como sendo de pares duplicados, $||V||$ é o número total de pares duplicados, $||C||$ é o número de pares candidatos retornados como resultado do processo de blocagem e $||T||$, o número de pares formados com todos os registros disponíveis.

Nos experimentos de escalabilidade, além dos critérios *PC* e *RR*, também observamos os custos de aprendizagem de máquina, que serviram para indicar os métodos que foram executados com menores custos de processamento. Nesses experimentos, consideramos os métodos mais eficientes como sendo aqueles que testam números menores de combinações de predicados até retornarem bons esquemas de blocagem como resultado de processamento em cada etapa. Os custos de aprendizagem de máquina verificados nos experimentos de escalabilidade podem ser calculados com a métrica CAM, que foi definida na Equação 2

Na métrica CAM, NCP representa o número de combinações de predicados verificados durante a execução dos métodos de blocagem, até que um esquema de blocagem

seja retornado como resultado em cada caso, e NP, o número de predicados disponíveis para a formação dos esquemas de blocagem.

Nos experimentos de qualidade, usamos as coleções *Cora*, *CiteSeer* e *Evolbase*. Nas verificações de escalabilidade, que foram feitas em seguida, usamos somente a coleção *Evolbase*, uma vez que esta é uma coleção sintética, podendo ser recriada com números diferentes de registros, como normalmente são as coleções usadas em experimentos de escalabilidade. As características das coleções usadas nos experimentos de qualidade são apresentadas na Tabela 2.

Coleção	Número total de registros	Tamanho das amostras de treino		
		Percentual de registros	Número de pares	Média de pares verdadeiros
Cora	1.295	5%	2.016	75
CiteSeer	154	30%	1.035	17
Evolbase	1.000	5%	1.225	5

Tabela 2. Amostras de experimentos de qualidade usando as coleções *Cora*, *CiteSeer* e *Evolbase*.

Nos dados dos experimentos de qualidade, usamos 5% do total de registros para compor as amostras de aprendizagem de máquina. Todas as coleções foram processadas por esse meio, com exceção da coleção *CiteSeer*, com a qual usamos amostras de 30% do total de registros, uma vez que em amostras de 5% encontramos poucos pares de registros duplicados, situação que prejudicaria o método *BSL Blocking*, visto que é baseado somente em exemplos de registros duplicados. Nesses dados, as amostras extraídas apresentaram médias de 75, 17 e 5 pares de registros duplicados das coleções *Cora*, *CiteSeer* e *Evolbase*, respectivamente. Para as verificações de escalabilidade, extraímos amostras de 5% do total de registros, usando arquivos de dados com o número de registros variando de 1.000 até 8.000 registros.

Como dito na Seção 3.1.3, o algoritmo de programação genética utiliza alguns parâmetros que conduzem sua execução. A Tabela 3 apresenta a configuração utilizada por nosso método.

Parâmetro	Valor usado
Número de gerações	5
Profundidade máxima da árvore de representação dos indivíduos	4
Número de indivíduos em cada geração	100
Número de melhores indivíduos copiados para geração posterior	4
Probabilidade de ocorrência de mutação	95%
Profundidade máxima em segmentos de mutação	3

Tabela 3. Valores de parâmetros usados com os métodos *BGP-SR*/*PR*.

Avaliando Qualidade e Escalabilidade

Na Tabela 4, apresentamos os resultados de experimentos feitos para a verificação de qualidade com as coleções *Cora*, *CiteSeer* e *Evolbase*.

Métodos avaliados	Cora		CiteSeer		Evolbase	
	PC (%)	RR (%)	PC (%)	RR (%)	PC (%)	RR (%)
BGP-SR	90,49	85,54	93,31	87,95	99,18	98,97
BGP-PR	94,72	93,54	91,02	93,02	98,47	98,46
DNF Blocking	92,20	48,57	91,47	82,42	84,08	99,81
BSL Blocking	86,51	39,74	85,15	65,55	92,87	88,59

Tabela 4. Resultados de experimentos de qualidade usando as coleções *Cora*, *CiteSeer* e *Evolbase*.

Na Figura 4, podemos verificar os índices de acerto conseguidos na detecção de pares de registros duplicados. Nesse caso, os menores percentuais de cobertura de pares duplicados (*PC*) foram os obtidos com o método *BSL Blocking*, possivelmente em razão do uso de exemplos limitados a pares duplicados de registros e, portanto, de números menores de exemplos quando os processos de aprendizagem de máquina foram realizados. Mesmo nesses casos, o percentual de pares de registros duplicados indentificados foi próximo de 85%, chegando a 92% nos experimentos feitos com a coleção *Evolbase*. Os métodos *BGP-SR*, *BGP-PR* e *DNF Blocking* apresentaram os melhores resultados, em particular os métodos *BGP-SR* e *BGP-PR* apresentaram o melhor desempenho quando avaliados com relação à redução de pares candidatos, o que atribuímos à maior eficácia do algoritmo de programação genética na combinação dos predicados de blocagem.

Nas Figuras 4(A), (B), (C) e (D), apresentamos os resultados dos experimentos de escalabilidade. Como podemos notar na Figura 4(A), os métodos *BGP-SR* e *BGP-PR* se mantiveram com boa cobertura de pares deuplicados de registros durante todas as etapas realizadas. Nas mesmas condições, o *DNF Blocking* apresentou resultados satisfatórios somente quando 4.000 ou mais registros foram usados nos experimentos.

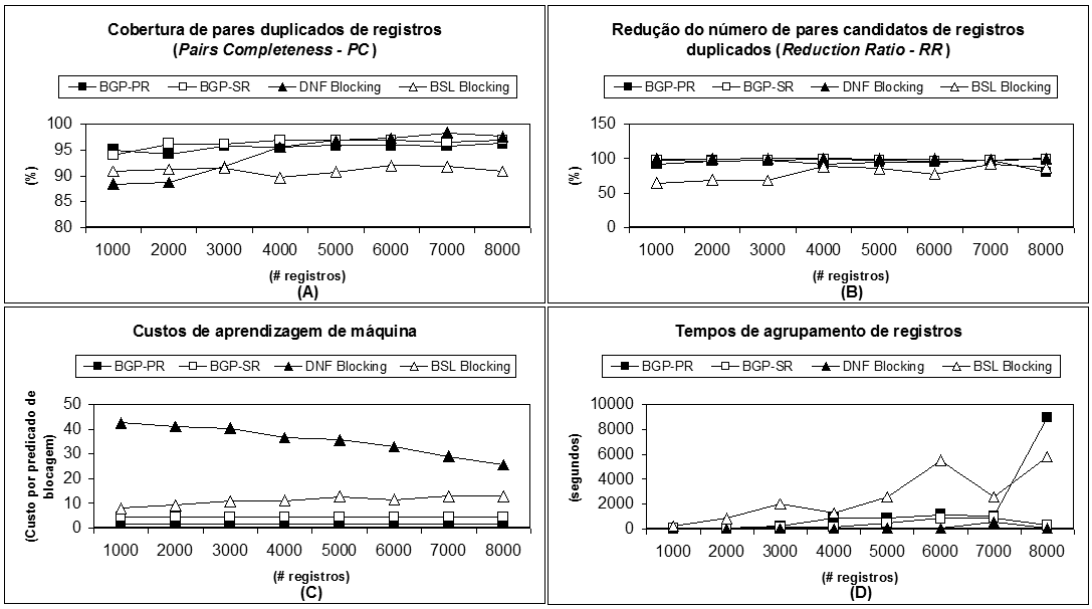

Figura 4. Verificação de escalabilidade dos métodos de blocagem.

Métodos iterativos como o *DNF Blocking* podem demandar muitos exemplos para encontrar bons esquemas de blocagem. No caso do *DNF Blocking*, a cada iteração do algoritmo principal, é retornada uma conjunção de predicados de blocagem. Por esse motivo, os esquemas de blocagem descobertos com esse método podem ser vistos como sequências de conjunções. Se poucos registros forem usados nos treinos de aprendizagem de máquina, esquemas apresentando poucas conjunções podem ser formados, possivelmente falhando na detecção de pares de registros duplicados em momentos posteriores.

Na Figura 4(B), podemos verificar as reduções conseguidas em termos do número de pares candidatos de registros. Como nas verificações anteriores, o método *BSL Blocking* foi o menos estável, possivelmente devido ao uso de quantidades menores de exemplos comparado aos demais métodos experimentados. Nos experimentos também medimos os custos de aprendizagem de máquina, que são apresentados na Figura 4(C), como forma de identificar os métodos que tiveram os menores custos de execução. Nessas verificações os métodos de programação genética foram os que apresentaram os menores custos, visto que foram as populações foram mantidas pequenas, com 500 esquemas de blocagem testados em cada fase de aprendizagem de máquina.

Nessas verificações de custo de processamento, o método *DNF Blocking* apresentou uma diminuição de custos, conforme o número de registros foi aumentado. Verificamos essa diminuição e notamos que ela ocorreu devido a predicados que foram descartados no início de cada etapa de execução.

Durante os experimentos de escalabilidade, também medimos os tempos de processamento em segundos de CPU quando os esquemas de blocagem foram usados para agrupar os registros. Esses tempos apresentaram variações significativas e por isso não os consideramos como referência para a verificação de escalabilidade dos métodos de blocagem.

Podemos observar esses tempos na Figura 4(D), como constatação do tempo necessário para o agrupamento dos registros em cada situação. Notamos que os melhores métodos foram o *DNF Blocking*, *BGP-SR* e *BGP-PR* até a etapa com 7.000 registros. Na etapa de 8.000 registros, o método *BGP-PR* apresentou tempos maiores de agrupamento de registros em razão de esquemas de blocagem formados por um número maior de predicados baseados em regras de *ngramas*. Essas regras podem atrasar o processo, uma vez que um número maior de blocos pode ser produzido quando os dados são processados, gerando muitos blocos semelhantes e, consequentemente, muitos pares de registros duplicados repetidos. Situações como essa podem ser justificadas nos casos em que a dificuldade de detecção de duplicatas de registros for um objetivo difícil a ser alcançado.

5. Conclusões e Trabalhos Futuros

Neste trabalho, propomos as técnicas de blocagem *BGP-SR* e *BGP-PR*, baseadas em programação génetica, que se mostraram as mais escaláveis no número de predicados que podem ser usados em processos de blocagem de registros, do que o observado em outros métodos recentes de blocagem. Usar maiores números de predicados é uma vantagem, uma vez que os processos são realizados com maior flexibilidade nesses casos e a identificação de registros duplicados pode ser ocorrer com maiores chances de acertos. Usando programação genética, resultados satisfatórios podem ser conseguidos sem que um número muito grande de combinações de predicados de blocagem seja analisado,

resultando em uma redução significativa nos custos de aprendizagem de máquina. Em trabalhos futuros os custos de processamento ainda podem ser reduzidos por meio de aprendizagem ativa, uma vez que passamos a usar amostras de dados menores nos processos de aprendizagem de máquina, diminuindo o tempo necessário para a verificação de cada combinação de predicados.

Agradecimentos

Este trabalho foi financiado parcialmente pelos projetos InfoWeb (550874/2007-0 CNPq), INCTWeb (573871/2008-6 CNPq), SIRIAA (55.3126/2005-9 CNPq); MinGroup (575553/2008-1 CNPq); Finep e Fapemig. Altigran Silva e Wagner Meira Jr. são bolsistas de produtividade do CNPq, Luiz Evangelista bolsista FAPEAM e Eli Cortez bolsista CAPES. O presente trabalho foi realizado com o apoio do UOL (www.uol.com.br), através do programa "UOL Bolsa Pesquisa" processo número 20090213165000.

Referências

Bhattacharya, I. and Getoor, L. (2004). Iterative record linkage for cleaning and integration. In *Proceedings of the 2004 ACM SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery (DMKD-2004)*, pages 11–18.

Bilenko, M., Kamath, B., and Mooney, R. J. (2006). Adaptive blocking: Learning to scale up record linkage. In *Proceedings of the 6th IEEE International Conference on Data Mining*, pages 87–96. ICDM-2006, IEEE, Inc.

Cohen, W. W., Ravikumar, P., and Fienberg, S. E. A comparison of string metrics for matching names and records. In *Proceedings of the Workshop on Data Cleaning, Record Linkage, and Object Consolidation*, pages 13–18. ACM SIGKDD 2003.

Cohen, W. W., Ravikumar, P., and Fienberg, S. E. (2003). A comparison of string distance metrics for name-matching tasks. In *Proceedings of the Workshop on Information Integration on the Web*, pages 73–78, Acapulco, Mexico. IJCAI-2003.

de Carvalho, M. G., Gonçalves, M. A., Laender, A. H. F., and da Silva, A. S. (2006). Learning to deduplicate. In *Joint Conference on Digital Libraries*, pages 41–50, New York, NY, USA. JCDL' 06, ACM.

Koza, J. R. (1998). *On the Programming of Computers by Means of Natural Selection*. The MIT Press.

McCallum, A. K., Nigam, K., and Ungar, L. (2000). Efficient clustering of high-dimensional data sets with application to reference matching. In *Proceedings of the 6th International Conference On Knowledge Discovery and Data Mining*, pages 169–178, Boston, MA. KDD-2000.

Michelson, M. and Knoblock, C. A. (2006). Learning blocking schemes for record linkage. In *Proceedings of the 21st National Conference on Artificial Intelligence (AAAI)*.

Sarawagi, S. and Bhamidipaty, A. (2002). Interactive deduplication using active learning. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 269–278, Edmonton, Alberta.

Winkler, W. (1994). Advanced methods for record linkage.

Winkler, W. E. (2006). Overview of record linkage and current research directions. Technical report, Statistical Research Division, U.S. Census Bureau, Washington, DC.