Uma Estratégia Eficiente de Treinamento para Programação Genética Aplicada a Deduplicação de Registros.

Davi Guimarães da Silva[1], **Moisés Gomes de Carvalho**[2], **Duivilly Brito**[2]

[1]Instituto Federal de Educação Ciência e Tecnologia do Pará (IFPA)

[2]Instituto de Computação - Universidade Federal do Amazonas (UFAM)

{davi.guimaraes}@ifpa.edu.br, {moises,db}@icomp.ufam.edu.br

Abstract. *Genetic Programming (GP) is a machine learning technique effectively used in the record deduplication problem. GP adopts a very expensive training step that requires that all records in a database be compared against each other several times. In this paper, we propose a novel approach for training step based on clustering technique combined with a sliding window. This combination aims at minimize the number of comparisons required in the training step without affecting its results. Our experiments using real datasets show that it is possible to reduce the time cost of the training step up to 72.8% compared to the GP state of the art approach without a significant impact in the quality of generated solutions.*

Resumo. *Programação Genética (PG) é uma técnica utilizada de forma eficaz na deduplicação de registros. Nela faz-se necessário realizar uma etapa de treinamento, em que cada registro é comparado com todos os outros na base de dados, tornando-a custosa. Neste artigo, propomos uma abordagem baseada na combinação de uma técnica de agrupamento e janela deslizante, visando minimizar a quantidade de comparações. Nossos experimentos com dados reais mostram que é possível reduzir o custo de treinamento da PG em até 72.8% comparado ao estado da arte sem uma redução significativa na qualidade das soluções geradas.*

1. Introdução

O problema de detecção e remoção de registros repetidos em um repositório é geralmente conhecido como deduplicação de registros [Koudas et al. 2006], que consiste em identificar e remover registros que são potencialmente os mesmos em uma base de dados.É uma tarefa complexa, que requer muito tempo e poder de processamento devido à grande quantidade de comparações necessárias para definir se um registro possui uma ou mais réplicas.

Em [Carvalho et al. 2008b], os autores apresentaram uma abordagem para a identificação de registros duplicados em repositórios recorrendo a uma técnica de Aprendizado de Máquina conhecida como Programação Genética (PG). Essa técnica apresenta alta precisão no processo de deduplicação em bases de dados com diferentes características. Porém, possui alto custo computacional da PG, tendo em vista que na etapa de treinamento, que visa "ensinar" a técnica identificar as características relevantes de boas soluções, cada registro é comparado com todos os outros registros, tornando pouco viável sua adoção.

Neste artigo propomos um método baseado na combinação de uma técnica de agrupamento e janela deslizante, para minimizar a quantidade de comparações exigidas na etapa de treinamento da PG. Os experimentos mostram que é possível tornar a etapa de treinamento da PG mais rápida mantendo o nível de qualidade das soluções, tendo em vista que técnica proposta obteve uma eficácia próxima da abordagem de [Carvalho et al. 2009] utilizado como *baseline*, com uma redução significativa no tempo de treinamento.

2. Trabalhos Relacionados

A tabela 1 apresenta abordagens utilizadas para o processo de deduplicação de registros com diferentes técnicas de aprendizagem de máquina.

Tabela 1. Abordagens aplicadas à deduplicação de registros.

Referência	Método Proposto
[Fellegi 1969]	Propôs um modelo baseado em probabilidades para otimizar a classificação dos pares de duas bases. Essa teoria é utilizada até os dias atuais
[Carvalho et al. 2008b]	Propõem uma abordagem baseada em PG que busca combinar evidências para a geração de funções de similaridade, utilizando uma pequena porção da base de dados para treino.
[Carvalho et al. 2008a]	Descrevem as principais propriedades da PG e apontam que a parametrização do algoritmo já proposto em [Carvalho et al. 2008b], pode alcançar resultados da deduplicação em até 30% melhores que anterior.
[Carvalho et al. 2009]]	Apresenta os resultados da PG aplicada a deduplicação de registro em três datasets: Cora, Restaurants e Synthetic obtendo melhor resultado que as abordagens anteriores.
[Carvalho et al. 2012]	Generalizam os resultados do método proposto, mostrando que a PG também é capaz de encontrar funções de deduplicação efetivas, mesmo quando as funções de similaridade não são conhecidas de antemão.
[Bianco; et al. 2013]	Propõem o arcabouço FS-Dedup para o processo de deduplicação de grandes volumes de dados, quando dependem de usuários especialistas para configurar as fases de blocagem e o algoritmo de classificação. Para isso, exploram algoritmos de deduplicação baseados em assinatura pela eficiência e escalabilidade.
[Ma et al. 2015]	Propõem uma nova nova abordagem para deduplicação de dados sem esquema e em paralelo utilizando MapReduce a partir de soluções que tornam o tamanho da janela deslizante adaptável, com a estratégia de múltiplas repetições com contagens adaptáveis visando acelerar o processo de deduplicação.
[Bianco; et al. 2016]	Os autores propõe uma estratégia de seleção de amostragem em duas fases, que vai desde a produção de pequenas subamostras aleatórias de pares candidatos em diferentes frações de conjuntos de dados e em seguida removem a redundância de pares para produzir um conjunto de treino menor e mais informativo.

É importante destacar que utilizamos a abordagem de [Carvalho et al. 2009] como *baseline*, por ser o mais recente com o uso da PG aplicada a deduplicação de registros e por isso os resultados são comparados com os deles. Além disso, os demais trabalhos da literatura utilizam outras técnicas para o mesmo problema.

3. Abordagem Proposta

A ideia desta proposta parte da abordagem de classificação que obedece as propriedades de transitividade das réplicas, em que, se um registro A for réplica de um registro B e B for réplica de um registro C, então A será réplica de C, e assim sucessivamente. Considerando essa transitividade, conclui-se que existe um nível de similaridade entre os registros réplicas disponíveis em um certo *dataset*. Assim, a nossa hipótese é que as réplicas terão distâncias equivalentes em relação ao um dado registro referência, a partir da similaridade entre cada registro, podendo diminuir a quantidade de comparação na fase de treinamento da PG.

3.1. Arquitetura geral da Abordagem Proposta

A Figura 1 apresenta uma visão geral da abordagem proposta com todas as etapas aplicadas à fase de treinamento da Programação Genética.

De modo geral, na **Etapa 1** ocorre a definição da estratégia de similaridade entre os registros de parte do *dataset*; na **Etapa 2** é realizada a comparação e ordenação dos registros em uma lista com base na similaridade retornada; na **Etapa 3** ocorre a definição dos parâmetros para a Janela Deslizante que será aplicada na lista ordenada; na **Etapa 4** é realizada a fase de treino da PG, e por fim os testes com parte do *dataset* não usada no treino para verificar quão boa foi a solução.

A abordagem proposta aplica os conceitos da técnica agrupamento baseada em particionamento de [Jain et al. 1999], em que os algoritmos de dividem os objetos entre os k *clusters* de acordo com a medida de similaridade adotada, de modo que cada objeto fique no

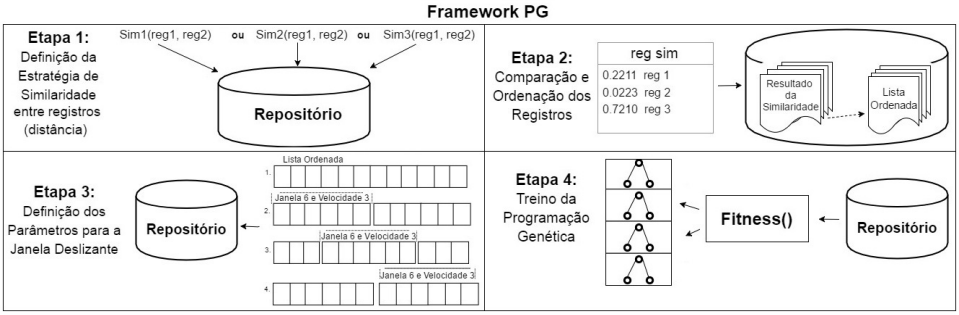

Figura 1. Visão geral da abordagem proposta.

cluster que forneça o menor valor de distância entre o objeto e o centro do mesmo. Assim, propomos os seguintes passos: 1) Escolher um registro *r* como o centro inicial do grupo; 2) Calcular a distância do registro *r* para cada um dos outros *n* registros utilizando uma função de similaridade; 3) Criar uma lista ordenada pelas distâncias retornadas pela função de similaridade; 4) Utilizar a abordagem de Janela Deslizante para comparar os registros que estiverem dentro do tamanho janela, obedecendo a lista ordenada anteriormente.

Tendo como base a classificação que obedece as propriedades de transitividade das réplicas, foram utilizadas três estratégias para escolha de um *registro referência*, isto é, o registro que será comparado com todos os outros do *dataset*, e que será usado com objetivo de definir a forma mais efetiva para realizar o agrupamento de registros similares.

Assim, foi testado o registro referência da seguinte forma: *1) Sintético:* É um registro criado a partir da junção entre atributos de registros diferentes existentes no *dataset* a ser avaliado. Esse registro deve conter todos os campos preenchidos. Desse modo, foi realizada a escolha de quatro registros aleatórios, visando a retirada de um atributo de cada para criar um novo registro. *2) Aleatório:* É escolhido um registro aleatoriamente dentre os registros existentes no *dataset* (sem ser previamente analisado). *3) String ruído:* Trata-se de um registro com as mesmas características do domínio dos tipos de dados existentes no *dataset*, contendo todos os campos preenchidos. O objetivo é representar o domínio dos tipos de atributo de uma determinada entidade, porém, não fica restrita aos valores do domínio, mas ao tipo de dados utilizado na representação daquele domínio.

Terminada essa etapa, os registros são ordenados por similaridade em ordem crescente. Para cada registro do *dataset* que possui réplica, é calculada uma distância desse registro para todas as sua(s) réplica(s). Assim, geramos as seguintes métricas: *Dmax*: Distância máxima encontrada entre um registro e qualquer uma de suas réplicas; *Dmed*: Distância média entre um registro e todas as suas réplicas. A partir desses valores, é calculada a média das distâncias máximas e a média das distâncias médias, para definir a função de similaridade que melhor agrupe um registro e suas réplicas, obtendo a menor *Dmax*. O objetivo é colocar os registros similares próximos uns dos outros.

3.2. Estratégia de Janela Deslizante

O próximo passo é comparar de todos os registros entre si. Para isso, foi adotada uma estratégia de janela deslizante proposta por [Ziv et al. 1977], na qual o autor propõe um modelo de "Janela Deslizante" de tamanho fixo ($w > 1$), que move-se sequencialmente sobre os registros ordenados a um deslocamento (v). Os registros que estiverem dentro da mesma janela serão comparados. Ao final, cada registro gerará ($2w$ - 1) pares para comparação, resultando um total de $O(nw)$ pares em um *dataset* com n registros no total.

A vantagem desta técnica é que a quantidade de pares comparados pode ser controlada e os registros que estiverem no intervalo definido pelo tamanho da janela, serão comparados. A partir da segunda execução os registros já comparados não serão mais comparados entre si, tendo impacto significativo no tempo de execução e reduzirá as comparações desnecessárias entre registros não similares.

3.3. Experimento para Definição da Função de Similaridade

A estratégia experimental de treinamento e testes utilizando PG adotadas neste trabalho, foram as mesmas do *baseline*, além das funções de similaridade e suas respectivas implementações: Softtfidf; Editdist; Jaro; Sortwinkler; Winkler; Bigrama; Bagdist; Seqmatch; e Compression. O objetivo foi encontrar uma função de similaridade que agrupasse registros similares mais próximos.

Em nossos experimentos utilizamos o *dataset Restaurants* [Bilenko 2003], que possui 864 registros divididos em 4 (quatro) *datasets* menores contendo 216 registros com o seguintes atributos: *(name, address, city, specialty)*. Na Tabela 2 são apresentados os resultados de *Dmax* e *Dmed* para cinco funções de similaridade que retornaram as menores distâncias no *dataset Restaurants*.

Tabela 2. Aplicação das funções de similaridade no *dataset Restaurants*.

Função	Distância *String* Ruido		Distância Registro Sintético		Distância Registro Aleatório	
	Dmax	Dmed	Dmax	Dmed	Dmax	Dmed
Editdist	**51**	**27**	62	29	52	31
Bagdist	61	35	64	36	61	34
Jaro	62	37	78	31	64	38
Sortwinkler	69	27	99	58	85	41
Bigram	72	33	155	46	75	36

Além dos resultados apresentados na tabela 2, também utilizaram-se as funções Softtfidf, Seqmatch, Winkler e Compression, que retornaram valores de *Dmax* e *Dmed* maiores para cada registro referência, porém *string ruido* sempre com distâncias menores. Assim, os experimentos mostraram que o uso do registro referência *string ruído* retorna os menores valores da *Dmax*, sendo aplicado para a avaliação do *dataset*.

3.4. Configuração Experimental para Janela Deslizante

Baseado nos resultados obtidos pela métrica *Dmed* no *dataset Restaurants* com o registro referência *string ruído*, foi definido para nossa experimentação utilizar janela deslizante inicialmente com aproximadamente metade do menor valor das *Dmed* obtidas no *dataset* e o tamanho do maior valor para janela um pouco acima do valor da média de todas as *Dmed* do *dataset*. Os valores do deslocamento foram baseados na porcentagem da quantidade de registros em cada *dataset*, ou seja, variaram em uma média de aproximadamente 3% a 10% em relação a quantidade de registros em cada *dataset*. Semelhantemente para os testes com as janelas utilizou-se três valores de deslocamento diferentes, descritos nas tabelas 3 e 4.

Para apresentar os resultados, criamos a Tabela 3 que exibe os resultados relacionados a Precisão, Revocação e F1 ([Baeza-Yates and Ribeiro-Neto 1999]), no treino e nos testes. A Tabela 4 apresenta os resultados do tempo gasto no treino (em segundos), a porcentagem do tempo em relação ao *baseline*, o total de comparações entre registros em cada configuração e a quantidade de réplicas identificadas corretamente. Para possibilitar a comparação da nossa abordagem com o *baseline*, criamos um campo descrito como ***"PG***

configuração padrão" no qual são apresentados os resultados obtidos, tanto para treino quanto para teste, tendo em vista que os experimentos reportados pelos autores não apresentaram os tempos de execução. Assim, foram executados nas mesmas condições de *hardware*, *software* e *dataset*, e serão descritos nas tabelas 3 e 4 a seguir.

Tabela 3. Resultados dos experimentos no *dataset Restaurants*

Configurações Gerais		Treino			Teste		
Parâmetros		Precisão	Revocação	F1 (σ)	Precisão	Revocação	F1 (σ)
PG configuração padrão		1.000	1.000	1.000 $\pm$(0.000)	1.000	0.963	0.981$\pm$(0.019)
Tam. janela	Deslocamento						
12	7	1.000	0.570	0.726 $\pm$(0.218)	1.000	0.535	0.694$\pm$(0.236)
23	7	1.000	0.801	0.889 $\pm$(0.100)	1.000	0.800	0.826$\pm$(0.109)
41	7	1.000	0.779	0.876 $\pm$(0.111)	1.000	0.713	0.858$\pm$(0.144)
	15	1.000	0.775	0.873 $\pm$(0.113)	1.000	0.750	0.856$\pm$(0.125)
58	7	1.000	0.953	0.976 $\pm$(0.024)	1.000	0.930	0.949$\pm$(0.036)
	15	1.000	0.810	0.895 $\pm$(0.095)	1.000	0.792	0.871$\pm$(0.105)
71	7	1.000	0.997	0.988 $\pm$(0.012)	1.000	0.949	0.975$\pm$(0.026)
	15	1.000	0.868	0.929 $\pm$(0.066)	1.000	0.829	0.898$\pm$(0.086)
	23	1.000	0.829	0.907 $\pm$(0.086)	1.000	0.802	0.889$\pm$(0.099)

Destaque-se que a precisão foi 1.0, isso significa apesar da redução na quantidade exemplos de treino, o método aprendeu com os exemplos existentes na janela dada. A tendência observada é que os valores de F1 melhoram com o aumento do tamanho da janela e a diminuição do valor de deslocamento desta. Ao realizar teste estatístico T com intervalo de confiança 95%, podemos identificar que muitas configurações de janela/deslocamento atingem desempenho de F1 estatisticamente equivalente à configuração padrão.

A Tabela 4 apresenta os resultados do tempo gasto na etapa de treinamento, a porcentagem comparada a *PG configuração padrão* e a quantidade de réplicas encontradas.

Tabela 4. Resultados dos experimentos no *dataset Restaurants*

Configurações (*Dataset* com 216 Registros)		Tempo do Treino (Segundos)		Total de Comparações	Réplicas Identificadas
(Total de réplicas no *dataset*: 10)		Total (seg)	%Tempo	Qtd (unid)	Qtd (unid)
PG configuração padrão		4466850	100%	23220	10
Tam. janela	Deslocamento				
12	7	14994	3.3%	1695	7
23	7	341260	7.6%	3860	8
41	7	68190	15.3%	7295	8
	15	592770	13.3%	6291	8
58	7	1095330	24.5%	10020	9
	15	881510	19.7%	9196	8
71	7	1214322	27.2%	11929	10
	15	1177405	26.4%	11046	9
	23	953950	21.4%	10675	9

Observando o resultado do tempo com a janela de tamanho 58 e deslocamento 7, o processo foi realizado em 24.5% do tempo comparado ao *PG configuração padrão*, encontrando 9 das 10 réplicas. Já a janela com tamanho 71 e deslocamento 7 realizou com 27.2% do tempo e encontrou todas as réplicas existentes. Houve uma redução de 72.8% do tempo gasto pela abordagem de [Carvalho et al. 2009] no treino da PG.

A partir dos resultados apresentados conclui-se que a tendência é que quanto maior o tamanho da janela e menor o deslocamento, os resultados são melhores. Quanto maior a janela, mais exemplos são utilizados no treinamento da PG. Já em relação ao deslizamento da janela, quanto maior o tamanho do deslocamento, as réplicas que poderiam estar em um determinado intervalo tendem a ficar em janelas diferentes e não serão comparados.

4. Conclusões e Trabalhos Futuros

Este trabalho propôs um método baseado na combinação de uma técnica de agrupamento e janela deslizante para a etapa de treinamento da PG, em que a partir da seleção de um registro referência, todos os outros registros foram comparados a ele por uma função de similaridade pré estabelecida, normalmente específica por base. Com os registros já agrupados com base nas distâncias retornadas pela similaridade, os mesmo foram comparados por uma estratégia de janela deslizante. A nossa combinação das técnicas, foi testada e comparada com os resultados do *baseline* e reforçaram a ideia de que a utilização dessa técnica permite a obtenção de resultados satisfatórios, com a vantagem da redução considerável no tempo de treinamento da PG, mantendo a qualidade dos resultados. Dessa forma, o diferencial desta proposta para o *baseline* é que sua aplicação foi específica para fase de treinamento da PG. Para trabalhos futuros planejamos realizar experimentos combinando as função de comparação entre os registros, utilizar paralelismo para a utilização da estratégia de janela deslizante, testar outros *datasets* de dados reais com diferentes domínios e graus de dificuldade, para ajudar a consolidar e estender os resultados obtidos neste trabalho.

Referências

Baeza-Yates, R. A. and Ribeiro-Neto, B. A. (1999). Modern Information Retrieval. *ACM Press/Addison-Wesley.*, New York, NY, USA. p. 39-48. (KDD 03).

Bianco;, G. D. et al. (2013). Tuning large scale deduplication with reduced effort. *In Proceedings International Conference on Scientific and Statistical Database Management, ACM, new york.*, p. 18:1-18:12. (SSDBM).

Bianco;, G. D. et al. (2016). A practical and effective sampling selection strategy for large scale deduplication. *IEEE International Conference on Data Engineering*, p. 1518-1519.

Bilenko, M.; Mooney, R. J. (2003). Adaptive Duplicate Detection Using Learnable String Similarity Measures. *In: ACM.*, New York, NY, USA. p. 39-48. (KDD 03).

Carvalho, M. G. et al. (2008a). The impact of parameter setup on a genetic programming approach to record deduplication. *S.B.C.; Brazilian Symp. Databases*, p.91-105.

Carvalho, M. G. et al. (2008b). Replica identification using genetic programming. *In: ACM Symposium on Applied Computing.*, p. 1801-1806.

Carvalho, M. G. et al. (2009). Evolutionary approaches to data integration related problems. *Tese. Universidade Federal de Minas Gerais*, p. 66-81.

Carvalho, M. G. et al. (2012). A genetic programming approach to record deduplication. *IEEE Transactions on Knowledge and Data Engineering; NJ, USA.*, v.24, p. 399-412.

Fellegi, I. P; Sunter, A. B. (1969). A theory for record linkage. *Journal of the American Statistical Association.*, [S.l.], v.64, n.328, p. 1183-1210.

Jain, A. K. et al. (1999). Data clustering: A review. *ACM Computing Surveys. 31(3)8*.

Koudas, N. et al. (2006). Record linkage: Similarity measures and algorithms. *ACM International Conference on Management of Data.*, p. 802-803, Chicago, USA.

Ma, K. et al. (2015). Large-scale schema-free data deduplication approach with adaptive sliding window using mapreduce. *The Computer Journal*, 58, n. 11, p.3187-3201.

Ziv, J. et al. (1977). A universal algorithm for sequential data compression. *IEEE Transactions on Information Theory*, 23(3) pp. 337-343.