Block Reordering: um algoritmo para evidenciar padrão Block em matrizes reordenáveis

André Matulionis dos Santos, Bruno Figueiredo Medina, Celmar Guimarães da Silva
Faculdade de Tecnologia – Universidade Estadual de Campinas
Limeira, Brasil

***Resumo*—Diferentes algoritmos permutam matrizes reordenáveis visando evidenciar visualmente padrões e permitir uma melhor análise de dados por parte de usuários. Este artigo propõe um algoritmo chamado Block Reordering, especializado em evidenciar o padrão Block em casos em que este padrão está oculto em uma matriz reordenável. Resultados de experimentos indicam que esse algoritmo obtém melhores resultados e em menos tempo que os algoritmos usados para comparação.**

Many algorithms permute reorderable matrices, in order to make visually evident patterns and to enable users to do better data analysis. This paper presents an algorithm called Block Reordering, which is specialized at make evident Block pattern when it is hidden inside a reorderable matrix. Experimental results point out that this algorithm produces better results than other algorithms considered at the experiment, and faster than them.

***Keywords*-padrões canônicos de dados; matrizes reordenáveis; visualização de informação**

I. Introdução

Visualização de dados é uma área de estudos que tem o objetivo de representar dados de forma visual e interativa para que sejam compreendidos intuitivamente por humanos. Dentro dessa área, Bertin [1] definiu e estudou o conceito de matriz reordenável, ou seja, uma matriz com linhas e colunas intercambiáveis. A matriz reordenável é um objeto base para o estudo da ordenação de matrizes, que visa aprimorar a visualização de dados de matrizes de valores, acarretando na descoberta de padrões significativos. Pode ser considerada como base para representações visuais como mapas de calor (*heatmaps*). Neste texto, matrizes reordenáveis e mapas de calor são tidos como conceitos equivalentes.

Wilkinson [2] definiu alguns padrões canônicos de matrizes com distribuições próprias de valores. Esses padrões formam a base para uma variedade de matrizes sintéticas que podem ocorrer no mundo real. Os algoritmos Multiple Binarization e Smoothed Multiple Binarization [3] foram desenvolvidos em nosso grupo de pesquisa na tentativa de evidenciar quatro desses padrões (Simplex, Equi, Circumplex e Band); porém não evidenciavam o padrão Block.

Este artigo propõe um algoritmo especializado na evidenciação do padrão Block quando este está presente, ainda que com ruído, em alguma permutação de linhas e colunas de uma matriz de dados. O algoritmo alcança essa evidenciação mesmo na presença de quantidade significativa de ruído em matrizes Block com colunas e linhas aleatoriamente permutadas.

Este trabalho está organizado da seguinte forma: o restante desta seção apresenta trabalhos relacionados a este e a definição do padrão Block; a Seção II apresenta o algoritmo Block Reordering proposto neste trabalho; a Seção III relata resultados de experimentos usados para comparar este algoritmo com outros da literatura; e por fim a Seção IV conclui este artigo e indica trabalhos futuros.

A. Trabalhos relacionados

Algoritmos de ordenação de matrizes são utilizados para facilitar visualizar informação que esteja potencialmente oculta em matrizes devido a permutações inapropriadas de suas linhas e colunas. Existem diversos algoritmos para ordenação de matrizes, cada um apresentando conceitos diferentes, visando gerar permutações apropriadas no menor tempo possível. Dentre os algoritmos encontrados na literatura, citamos os utilizados nesta pesquisa:

- MDS (Multidimensional Scaling) [4]: conjunto de técnicas para visualização de informação, que procuram projetar um conjunto de tuplas n-dimensionais em um espaço p-dimensional, onde $p << k$. Usando $p = 1$, MDS pode ser usado para gerar uma ordem entre linhas e entre colunas de uma matriz de dados, com base em suas matrizes de similaridade.
- 2D-Sort [5]: algoritmo baseado em heurísticas. Iterativamente procura prover uma ordem entre linhas e entre colunas, de forma alternada.
- Multiple Binarization (*MB*) [3]: algoritmo inspirado em ordenação de matrizes binárias por árvores PQR [6]. De forma geral, o método consiste em (1) verificar qual o padrão canônico possivelmente presente na matriz de dados fornecida, (2) escolher valores delimitadores (*thresholds*) conforme o padrão escolhido, (3) criar múltiplas matrizes binárias (conforme os delimitadores escolhidos) com base na matriz de dados, e (4) usar sobre essas matrizes a ideia básica de reordenação do algoritmo PQR-Sort [6], refletindo essa reordenação sobre a matriz de dados original.
- Smoothed Multiple Binarization (*SMB*) [3]: aprimoramento do algoritmo MB para melhor lidar com matrizes com ruídos. Utiliza suavização (*smoothing*) como passo intermediário na geração da matriz ordenada. De forma resumida, o SMB faz primeiramente uma ordenação similar à do MB, seguida por um passo de suavização e uma outra ordenação. A ordem calculada para linhas

e colunas é replicada na matriz original, descartando a matriz suavizada.

B. Referencial teórico

Sendo o padrão Block o foco desta pesquisa, esta seção o define em detalhes. Através de linhas horizontais e verticais, é possível particionar uma matriz de dados em submatrizes, chamadas de blocos. Uma matriz em padrão Block é uma matriz particionada em blocos, sendo que em cada bloco há um valor principal, compartilhado pelas células daquele bloco (excetuando aplicação de ruído).

No exemplo da Figura 1, os blocos foram calculados com o seguinte padrão de bits linha-a-linha, sem ruído:

$$\{000, 100, 010, 110, 001, 101, 011, 111\} \tag{1}$$

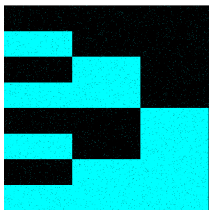

Figura 1. Exemplo de matriz block.

As linhas e colunas dos blocos são calculados através de um número arbitrário de bits k que afeta ambas as dimensões. O padrão anterior tem $k = 3$, considerando 3 bits para cada linha. Cada bloco da matriz tem como dimensões:

$$B_h = n/2^k \tag{2}$$

e

$$B_w = p/k, \tag{3}$$

onde B_h e B_w são o número de linhas e colunas do bloco, respectivamente; n e p são o número de linhas e colunas da matriz, respectivamente; e k é um número arbitrário pelo qual é computado o padrão linha-a-linha.

II. Algoritmo Block Reordering

Este artigo propõe um novo algoritmo, denominado Block Reordering (BR). A proposta se baseia na suposição de que a matriz a ser ordenada é uma matriz pré-Block, ou seja, uma matriz para a qual haja uma permutação apropriada de linhas e colunas que a transforme em uma matriz Block. Este artigo não oferece uma maneira de se verificar se a matriz informada é ou não pré-Block, sendo este aspecto relegado a trabalhos futuros. Contudo, uma vez havendo uma sinalização de que a matriz em questão é pré-Block, o algoritmo BR pode ser aplicado para uma evidenciação desse padrão.

O algoritmo BR se baseia no princípio de que todas as colunas semelhantes devem estar em conjunto, e o mesmo é válido para as linhas. Ele tem como vantagem a qualidade de reordenação em matrizes Block, mesmo em casos de taxa de ruído altas, e a velocidade com que é executado. Como desvantagem, ele não considera outros padrões de dados em matrizes.

A. Conceitos

O algoritmo BR utiliza conceitos próprios que precisam ser definidos antes de se apresentar o algoritmo em si.

1) Cálculo de Ruído da Coluna: Para calcular a quantidade de ruído de uma dada coluna, teve-se como base a suposição de que colunas do padrão Block sem ruído têm aproximadamente a mesma quantidade de valores 0 e 1, ou seja, metade dos valores tende a ser 0, e a outra metade tende a ser 1. Para casos em que o valor está no intervalo entre 0 e 1, o algoritmo o arredonda para o valor mais próximo. Matrizes com dados escalares podem ser normalizadas para o intervalo de 0 a 1, recaindo no caso anterior. Deste modo, a taxa de ruído é dada por:

$$U_c = \frac{|Q - N/2|}{N}, \tag{4}$$

onde U_c = Taxa de ruído da coluna c; Q = Quantidade de dados da coluna com um valor específico; e N = Quantidade de linhas da coluna.

Nota-se que como a fórmula utiliza valores absolutos, Q pode ser a quantidade de zeros ou a quantidade de uns.

2) Colunas Irmãs: Colunas irmãs são aquelas que devem estar próximas umas às outras, e que, se o ruído dos dados fosse desconsiderado, seriam iguais.

3) Coluna Pivô: Colunas pivô são colunas da matriz usadas pelo algoritmo para agrupar colunas semelhantes a elas (colunas irmãs). Para escolher as colunas pivô, o algoritmo deve considerar colunas com menor ruído dentre as disponíveis em uma dada iteração do algoritmo.

4) Coluna Ótima: A coluna ótima é uma coluna não inclusa na matriz, pois é criada a partir dos dados da matriz. Ela é uma coluna que representa um dado agrupamento de colunas similares, porém não possui ruído.

B. Algoritmo

O algoritmo BR é apresentado na Figura 2. Ele assume como entrada uma matriz M, pré-Block, a ser reordenada.

Cada etapa do algoritmo BR é detalhada a seguir.

$O \leftarrow \emptyset$ {Lista de colunas ótimas}
$C \leftarrow$ lista de colunas da matriz M
Enquanto $C \neq \emptyset$ **faça**
  $p \leftarrow$ Obtenha coluna pivô de C
  Reordene C com base na similaridade com p
  $I \leftarrow$ colunas irmãs de p
  $o \leftarrow$ coluna ótima, baseada em I
  Adicione o em O
  Retire de C os elementos pertencentes a I
Fim (Enquanto)
$M' = M$
Reordene as colunas de M' com base em C
Reordene as linhas de M' com base em O
Retorne M'

Figura 2. Block Reordering algorithm

1) Obter Coluna Pivô: O pivô escolhido é a coluna com a menor taxa de ruído dentre as colunas não processadas pelo algoritmo.

2) Reordenar com Base na Similaridade: É calculada a similaridade (via coeficiente Simple Matching) entre a coluna pivô atual e as colunas em C. Em seguida, o algoritmo ordena essas colunas de forma decrescente de acordo com a similaridade calculada.

3) Selecionar Colunas Irmãs: Para selecionar as colunas irmãs da coluna pivô, o algoritmo faz uma busca sequencial por todas as colunas com similaridade maior que um índice específico definido arbitrariamente. Colunas com similaridade maior que este limite são consideradas irmãs (e, como exposto anteriormente, devem estar próximas umas às outras).

A implementação proposta definiu esse limite como 60%, visando desconsiderar ruídos que possam diminuir a similaridade entre colunas irmãs bem como reduzir a quantidade de falsos positivos.

4) Criar Coluna Ótima: A coluna ótima referente a um conjunto de colunas irmãs é criada com base nos valores dessas colunas. O valor de uma célula da coluna ótima é o valor que mais ocorrer nas células respectivas das colunas irmãs.

Esse método de criação da coluna ótima tem a limitação de que o índice de ruído seja menor que 50%, pois para índices maiores o valor da maioria das colunas irmãs sofre alterações, e isso se reflete na reordenação final.

5) Reordenação final: Para reordenar a matriz, o algoritmo toma como base as colunas irmãs e as colunas ótimas. Para reordenar as colunas da matriz, as colunas irmãs são agrupadas entre si, seguindo a ordem de seleção das colunas pivô. As colunas são ordenadas da coluna com menor taxa de ruído até a coluna com maior taxa de ruído, da esquerda para a direita.

Para reordenar as linhas da matriz, o algoritmo usa como base as colunas ótimas. Para cada coluna ótima, ele ordena as linhas da coluna em ordem crescente, e em seguida aplica essa ordem às linhas da matriz. Após fazer o processo para todas as colunas ótimas, estas tendem a apresentar o padrão Block, bem como a matriz de dados.

III. Experimentos e resultados

O algoritmo proposto foi validado por dois experimentos, cujas descrições e resultados são apresentados a seguir.

Experimento 1: Para analisar o algoritmo BR em termos de qualidade da reordenação e de tempo de execução, foi usada a funcionalidade de execução de experimentos de comparação de algoritmos de reordenação de matrizes, provida pela ferramenta MRA [6]. O experimento foi projetado da seguinte maneira:

- Foram utilizadas 100 matrizes de dimensão 300×300 com padrão Block para cada configuração, devidamente permutadas de forma aleatória;
- As seguintes configurações foram utilizadas:
  - Taxas de ruídos: 5%, 10%, 15%, 20%, 25%, 30%, 35%, 40%, 45%, 50%;
  - Algoritmos de reordenação: 2D Sort, MDS, MB, SMB, BR;
  - Valores para parâmetro k do padrão Block: 3, 5 e 7;
- Função de avaliação: Minimal Span;
- Coeficiente de dissimilaridade: Distância euclidiana.

Foi escolhida a função de avaliação Minimal Span (MS) [7] porque se deseja que colunas (e linhas) parecidas estejam agrupadas. Associada a ela, foi usada distância euclidiana como coeficiente de dissimilaridade, pois é comumente utilizada para cálculo da distância entre tuplas.

Cada matriz produziu três fatores de avaliação: o valor de MS das linhas e das colunas, e o tempo de execução. As avaliações finais são as médias das avaliações das 100 matrizes de entrada. Os resultados de valor menor são os considerados com melhor qualidade.

Optamos por apresentar neste artigo apenas os resultados para $k = 5$, sumarizados nas Figuras 4, 5 e 6. Nessas figuras, verifica-se que o algoritmo BR apresenta os melhores valores da função MS dentre os algoritmos testados, tanto para linhas quanto para colunas, em especial quando o ruído é maior que zero. Adicionalmente, o algoritmo BR se executa em tempo mais rápido que os demais; seu tempo de execução é inferior a 250 ms. Resultados similares são obtidos com $k = 3$ e $k = 7$.

Apesar de terem sido testadas apenas matrizes com dimensão 300×300 e taxas de ruído específicas, considera-se que os resultados sejam similares em outras dimensões de matrizes e com outras taxas de ruído.

Experimento 2: O segundo experimento visou exemplificar as saídas dos algoritmos 2D Sort, MB, MDS, SMB e BR. A Figura 3 apresenta os resultados de execuções desses algoritmos sobre uma matriz padrão Block, com $k = 5$ e diferentes taxas de ruído. Nota-se que na ausência de ruído a maioria dos algoritmos faz algum agrupamento, porém falha ao tentar reproduzir o padrão. O BR, contudo, recompõe o padrão, mas de forma espelhada no eixo horizontal. Isso ocorre devido à reordenação de linhas, que utiliza as colunas ótimas como base da reordenação da direita para a esquerda e prioriza as colunas ótimas à esquerda. Na presença de ruído, os algoritmos testados não produzem resultados que evidenciem qualquer padrão. Novamente, a exceção é a saída do BR, em que grande parte do padrão Block é perceptível, falhando apenas em um conjunto isolado de colunas à direita da matriz.

IV. Conclusão

Neste artigo, foi apresentado o algoritmo Block Reordering para reordenação de matrizes pré-Block. Mostrou-se que ele é particularmente apropriado para evidenciar padrões Block em matrizes pré-Block com diferentes níveis de ruído (de 0 a 50%).

Trabalhos futuros incluem desenvolver um algoritmo para estimar se uma dada matriz de dados possui o padrão pré-Block, bem como incluir o algoritmo BR como opção a ser utilizada junto aos algoritmos SMB e MB na presença de uma matriz pré-Block.

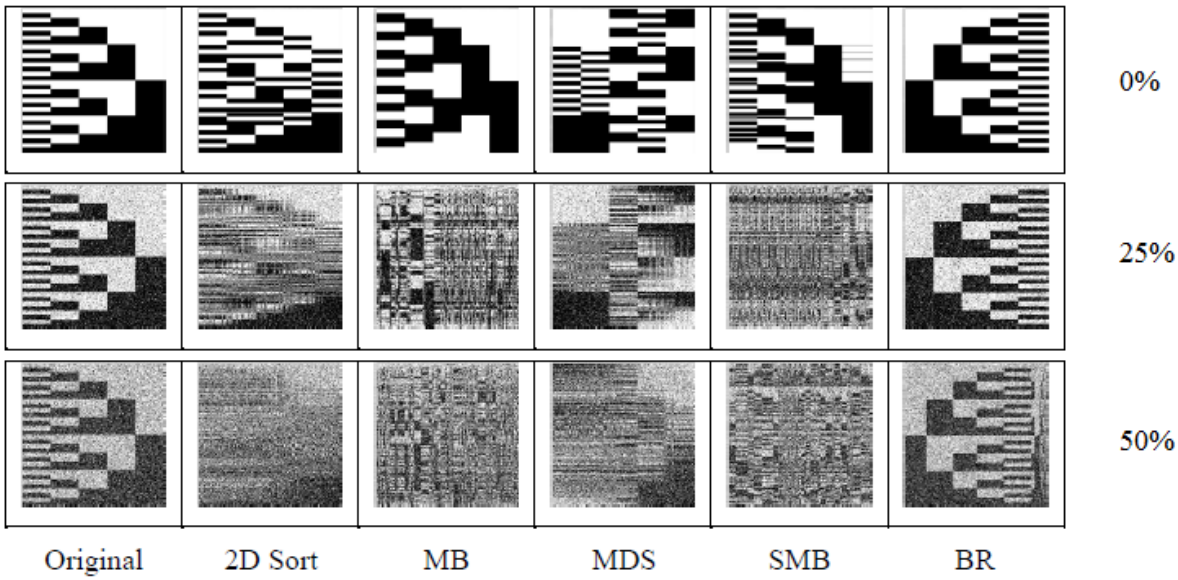

Figura 3. Exemplos de saídas de diferentes algoritmos de ordenação. Primeira coluna: matriz original; demais colunas: saídas dos algoritmos de ordenação ao receberem a matriz original já embaralhada. As linhas indicam diferentes níveis de ruído na matriz original (0 a 50%).

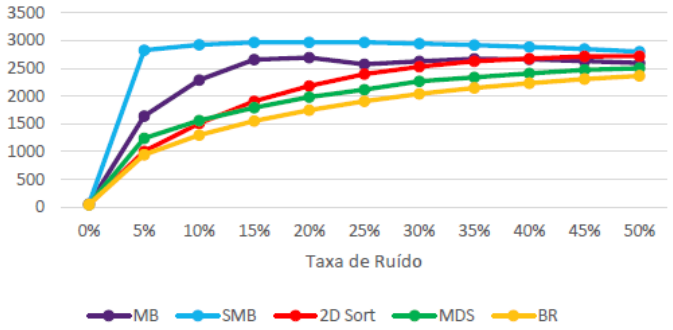

Figura 4. Média de valores da função MS relativa às colunas das matrizes reordenadas obtidas pelos algoritmos.

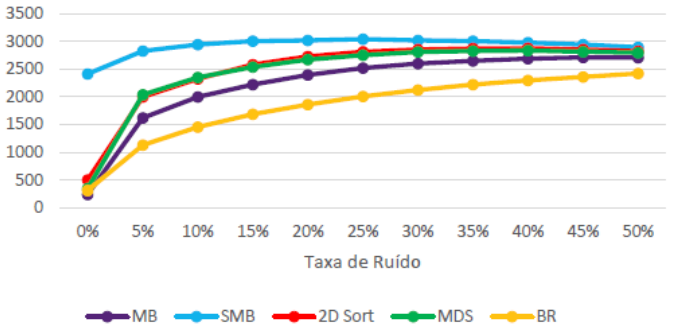

Figura 5. Média de valores da função MS relativa às linhas das matrizes reordenadas obtidas pelos algoritmos.

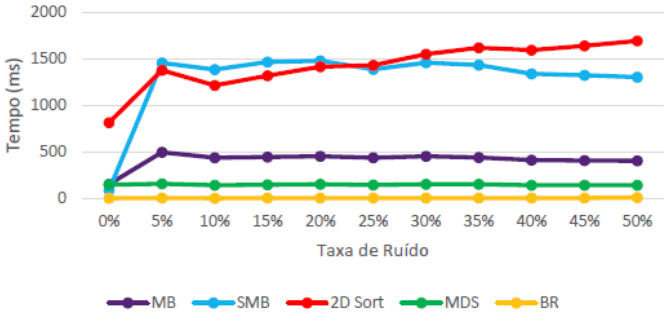

Figura 6. Média de tempo de execução dos algoritmos.

AGRADECIMENTOS

Os autores gostariam de agradecer ao financiamento desta pesquisa pela Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP), processo nº 2015/14854-7, e pela Pró-Reitoria de Pesquisa da Universidade Estadual de Campinas.

REFERÊNCIAS

[1] J. Bertin, *Semiology of Graphics: Diagrams, Networks, Maps*. University of Wisconsin Press, 1983.

[2] L. Wilkinson, *The Grammar of Graphics*. Springer Science & Business Media, 2005.

[3] B. F. Medina, "Using PQR tree for quantitative data matrix reordering." Master's thesis, University of Campinas, 2015.

[4] B. J. Kruskal and M. Wish, *Multidimensional Scaling*. Sage Publications, 1978.

[5] E. Mäkinen and H. Siirtola, "Reordering the reorderable matrix as an algorithmic problem," in *Theory and Application of Diagrams*. Springer, 2000, pp. 453–468.

[6] C. G. Silva, M. F. Melo, F. P. Silva, and J. Meidanis, "PQR sort: using PQR trees for binary matrix reorganization," *Journal of the Brazilian Computer Society*, vol. 20, no. 1, 2014.

[7] H.-M. Wu, S. Tzeng, and C.-h. Chen, "Matrix Visualization," in *Handbook of Data Visualization*. Springer, 2008, pp. 681–708.