Heuristics for a Matrix Symmetrization Problem*

EL AFRIT Mohamed Amine

January 2009

Abstract

The problem considered in this document is: let a square, non symmetric, (0, 1)-matrix, find a permutation of its columns that yields a zero-free diagonal and maximizes the symmetry. The problem is known to be *NP-hard*. A fast iterative-improvement based heuristic is proposed and the performance of the heuristic will be evaluated on a large set of matrices.

1 Context

In this work focuses on the optimization version of the matrix symmetrization with zero-free diagonal problem. The problem arises in a preprocessing phase of some other algorithms. For example, when a given sparse matrix A has an unsymmetric pattern, most of the graph partitioning and ordering algorithms are applied to the pattern of the symmetric completion $A + A^T$ [2] [3] [4]. Parallel solvers use parallel tree which are defined only for a symmetric pattern matrices, we also need to have a symmetric matrix [3] for this kind of problem.

The techniques proposed in this work can be used to make a given matrix more symmetric and obtain a sparser symmetric completion. In other words, the proposed techniques can help to improve the running time of the aforementioned algorithms and their solutions' quality.

Since the decision problem is *NP-complete*, and hence the optimization version of the symmetrisation problem is *NP-hard*, the solution that is proposed here is a heuristic algorithm. The heuristic is tested on a large set of matrices [5].

2 Some definitions and well-known results

2.1 Definitions

Definition 2.1. Bipartite graph associated to a matrix [6]

A bipartite graph (or bigraph) is a graph whose vertices can be divided into two independent disjoint sets R and C such that every edge connects a vertex in R to one in C.
Here the two sets R and C correspond, respectively, to the rows and the columns of out initial matrix to be symmetrized.
A bipartite graph $G = (R \bigcup C, E)$ is associated to the matrix A that we want to symmetrize, where R and C are the two sets in the vertex bipartition, and E is the set of edges. Here, the vertices in R and C correspond, respectively, to the rows and the columns of A such that $(r_i, c_j) \in E$ if and only if $a_{ij} = 1$. Although the edges are undirected, we will always specify an edge $e \in E$ as $e = (r_i, c_j)$; the first vertex will always be a row vertex and the second one will always be a column vertex. An edge $e = (r_i, c_j)$ is called to be incident on the vertices r_i and c_j

Definition 2.2. Notation

$N(r_i)$ denotes the neighbors of a row vertex r_i,
$d(.)$ denotes the degree of a vertex, e.g., $d(r_i) = |N(r_i)|$;
w_{ij} denotes the weight of an edge (r_i, c_j).

*This note is a summary of Bora Uçar publication entitled "*Heuristics for a Matrix Symmetrization Problem*" [1]. This work was supported by "Agence Nationale de la Recherche", ANR-06-CIS6-010.

Definition 2.3. *M-alternating* cycle

| Given a matching M, an *M-alternating* cycle is a simple cycle whose edges are alternately in M and not in M,

Definition 2.4. The symmetry score

For a given square (0,1)-matrix A, the symmetry score is defined as $S(A) = \sum_{a_{ij} \neq 0} a_{ij}.a_{ji}$

2.2 Well-known results

1. An even cycle contains an even number of vertices,
2. A set of edges M is a matching if no two edges in M are incident on the same vertex,
3. A matching is called perfect if for any vertex v in G, there is an edge in M incident on v,
4. *mate(v)* is used to denote the vertex matched to the vertex v in a matching M,
5. A perfect matchings in the bipartite graph G correspond to permutations which yield zero-free diagonals we define the permutation matrix M as $m_{ij} = 1$ if $(r_i, c_j) \in M$ and 0 otherwise [7],
6. Let M^* and M be, respectively, an optimal perfect matching maximizing the symmetry score, and another perfect matching on the bipartite graph G. Then the symmetric difference $M \bigoplus M = (M \setminus M) \bigcup (M \setminus M)$ contains only isolated vertices and even cycles,
7. Let C4 be the set of unique alternating cycles of length four, then the score will be $S(AM) = n + 2 \times |C4|$
8. A vertex v can be in at most $d(v) - 1$ alternating cycles.
9. For any matching M with $(r_i, c_j) \in M$, the contribution of the matching edge (r_i, c_j) to $S(AM)$ is limited by $min\{d(r_i), d(c_j)\}$

3 The heuristic

This heuristic works on the bigraph representation of a matrix. It starts with a perfect matching to guarantee a zero-free diagonal, and then iteratively improves the current matching to increase the symmetry while maintaining a perfect matching at all times.

Algorithme 1 Compute the symmetry score

Require: a bipartite graph $G = (R \cup C, E)$ corresponding to an $n \times n$ matrix A
Require: a perfect matching
Ensure: $score = S(AM)$

```
mark(r) ← 0 for all r ∈ R
score ← 0
for all (r_i, c_j) ∈ M do
  for all c ∈ N(r_i) do
    mark(mate(c_i)) ← j
  end for
  for all (r_i, c_j) ∈ M do
    if mark(r) = j then
      score ← score + 1
    end if
  end for
end for
```

Consider two matching edges (r_i, c_j) and (r_k, c_l) such that $(r_i, c_l) \in E$ and $(r_k, c_j) \in E$. These four edges form an *M-alternating* cycle of length four.

The refinement process is organized in passes (such as that in [**?**]). At each pass, we build the set *C4* of unique alternating cycles of length four using an algorithm much like Algorithm 1

We visit the unique alternating cycles of length four in a random order. Among the cycles those whose vertices are not affected by a previous operation and **with a non-negative effect on symmetry score are reversed** i.e; for example the alternating cycle $\{(r_i, c_j)\ ,\ (r_k, c_l)\}$ will be $\{(r_i, c_l)\ ,\ (r_k, c_j)\}$

Algorithme 2 Refine a perfect matching

Require: a bipartite graph $G = (R \cup C, E)$ corresponding to an $n \times n$ matrix A
Require: a perfect matching
Ensure: an other perfect matching $M^{(1)}$ where $S(AM^{(1)}) \geq S(AM^{(0)})$
  $M^{(1)} \leftarrow M^{(0)}$
  $C4 \leftarrow (r_1, c_1, r_2, c_2)$: $(r_1, c_1) \in M^{(1)}$ *and* $(r_2, c_2) \in M^{(1)}$ *and* $(r_1, c_2) \in E$ *and* $(r_2, c_1) \in E$
  while $C4 \neq 0$ **do**
    pick a cycle $C = (r_1, c_1, r_2, c_2) \in C4$
    if isReversible(C) **and** $gain(C) \geq 0$ **then**
      $M^{(1)} \leftarrow M^{(1)} \bigoplus C$
    end if
    remove the cycle C from C4
  end while

The test *isReversible(C)* returns true if the current matching contains the edges (r_1, c_1) and (r_2, c_2).

The gain is computed by using the main "for loop" of Algorithm 1 for the edges (r_1, c_1) and (r_2, c_2), and then for the edges (r_1, c_2) and (r_2, c_1). The difference between the returned scores gives the gain of reversing the edges in the cycle C.

3.1 Deficiencies

1. The set of alternating cycles of length four, *C4*, is constructed according to the initial matching,
2. Each cycle in *C4* is considered at most once,
3. Due to the nonnegative gain requirement, the algorithm cannot escape from a local minimum.

3.2 Ameliorations

Having observed these deficiencies, two alternatives were designed :

First alternative The first alternative starts with the same set *C4* as in Algorithm 2 with more involved data structures and operations. It maintains *C4* as a priority queue using the gain of a cycle as its key; tentatively modifies the current matching along the best cycle, even along those with a negative effect; at the end, realizes the most profitable prefix of modifications.

⇒ This approach obtained better results than Algorithm 2, with increases in running time.

Second alternative The second alternative visits the row vertices in a random order, computes the best length four alternating cycle containing that vertex, and modifies the current matching along that cycle if the gain is nonnegative.

⇒ Algorithm 2 outperformed this alternative in terms of solution quality.

3.3 Initialization of the algorithm

Here the aim is to find an upper bound on the attainable symmetry score.
Consider a maximum weight perfect matching M^*_{B1} , subject to edge weights $w_{ij} = min\{d(r_i), d(c_j)\}$, in the bipartite graph G. The weight $w(M^*_{B1})$ defines an upper bound, referred to as *UB1*.

We can obtain an improved upper bound by observing that all neighbors of the vertex r_i may not be matchable to the neighbors of c_j two vertices r_i and c_j can contribute at most by the weight of the maximum weight matching M^*_{ij} (with unit edge weights), in the induced subgraph $G_{ij} = (N(c_j) \bigcup N(r_i), E \bigcap N(c_j) \times N(r_i))$. Consider a maximum weight perfect matching *MB2*, subject to edge weights $w_{ij} = w(M^*_{ij})$, in G. The weight $w(MB2)$ defines an upper bound, referred to as *UB2*.

⇒ Both of the perfect matchings *MB1* and *MB2* can be used as an initial solution. we have $w(M^*_{B2}) \leqslant w(M^*_{B1})$.

It is better to use M^*_{B1} as an initial solution because the CPU cost of finding M^*_{B2} can be very high.

4 Experiments

The heuristic was tested with two sets of square matrices from University of Florida Sparse Matrix Collection [5].

Matrices in the first set originally have symmetric nonzero pattern and full sparse rank, and they satisfy the following assertions regarding the order n and the number of nonzeros nnz:

- $n \geqslant 1000$,
- $nnz \leqslant 106$,
- $nnz \geqslant 3 \times n$.

There were a total of 420 such matrices

The original matrices have zero-free diagonal. If B is the pattern of an original matrix, then the corresponding matrix A in this set has the pattern of $P(B + I)Q$, where I is the $n \times n$ identity matrix, and P and Q are random permutation matrices.

The matrices in the second set are the 28 public domain matrices used in [7]. These matrices are highly unsymmetric. For these 28 matrices, there are zeros in the main diagonal and the optimal symmetry score is not known.

In the experiments, initial matching $M^{(0)}$ is chosen to be M^*_{B1} , the perfect matching that defines the upper bound *UB1*.

Most of the improvements are obtained within the first few passes. In practice the number of refinement passes is limited to five, i.e., Algorithm 2 is applied five times.

The average symmetry score 0.628 of $M^{(0)}$ is improved by around 38% in five refinement passes, resulting in an average symmetry score of 0.872 for $M^{(5)}$.

5 Conclusion

This work proposed a heuristic to solve the matrix symmetrization with zero-free diagonal problem. The heuristic starts from a judiciously chosen initial solution and iteratively improves it. Experiments were presented on two sets of matrices. The solutions found by the proposed heuristic are, on average, around 0.87 of the exact solutions for the 420 matrices

Acknowledgments

The analysis of this article gave me the apportunity to have an experience in the domain of researching in computer science.
I thank Pierre Ramet and Bora Uçar because they helped me to understand and to sum up this article.

References

[1] Bora Uçar. *Heuristics for a Matrix Symmetrization Problem.* CERFACS 42 Avenue Gaspard Coriolis, 31057, Toulouse, Cedex 1, France (ubora@cerfacs.fr). `http://graal.ens-lyon.fr/~bucar/abstracts/UcarSymmetrize.html`.

[2] T.G Hendrickson, B.Kolda. Partitioning rectangular and structurally unsymmetric sparse matrices for parallel processing. Technical report. SIAM Journal on Scientific Computing.

[3] P. Ramet P. Hénon and J. Roman. *A High-Performance Parallel Direct Solver for Sparse Symmetric Definite Systems.*

[4] E. Lawler. Combinatorial optimization: Networks and matroids. Technical report. Dover, Mineola, New York (unabridged reprint of Combinatorial Optimization: Networks and Matroids, originally published by New York: Holt, Rinehart, and Wilson, c1976) (2001).

[5] T.Davis. University of florida sparse matrix collection na digest 92/96/97 (1994/1996/1997). Technical report. `http://www.cise.ufl.edu/research/sparse/matrices`.

[6] web. Cours et notions sur les graphes. Technical report. `http://fr.wikipedia.org` ...

[7] Koster J. Duff, I.S. On algorithms for permuting large entries to the diagonal of a sparse matrix. Technical report, SIAM Journal on Matrix Analysis and Applications 22, 973-996 (2001).