Recognizing Sparse Perfect Elimination Bipartite Graphs

Matthijs Bomhoff

University of Twente
The Netherlands

CSR 2011, June 18

Gaussian Elimination Example

$$\begin{pmatrix} -24 & 7 & 4 & -6 \\ -8 & 2 & 0 & 0 \\ 5 & 0 & 4 & 0 \\ 1 & 0 & 0 & 3 \end{pmatrix}$$

Gaussian Elimination Example

$$\begin{pmatrix} -24 & 7 & 4 & -6 \\ -8 & 2 & 0 & 0 \\ 5 & 0 & 4 & 0 \\ 1 & 0 & 0 & 3 \end{pmatrix} \quad \begin{pmatrix} -24 & 7 & 4 & -6 \\ 0 & \neq 0 & \neq 0 & \neq 0 \\ 0 & \neq 0 & \neq 0 & \neq 0 \\ 0 & \neq 0 & \neq 0 & \neq 0 \end{pmatrix}$$

Gaussian Elimination Example

$$\begin{pmatrix} -24 & 7 & 4 & -6 \\ -8 & 2 & 0 & 0 \\ 5 & 0 & 4 & 0 \\ 1 & 0 & 0 & 3 \end{pmatrix} \quad \begin{pmatrix} -24 & 7 & 4 & -6 \\ 0 & \neq 0 & \neq 0 & \neq 0 \\ 0 & \neq 0 & \neq 0 & \neq 0 \\ 0 & \neq 0 & \neq 0 & \neq 0 \end{pmatrix}$$

$$\begin{pmatrix} -24 & 7 & 4 & -6 \\ -8 & 2 & 0 & 0 \\ 5 & 0 & 4 & 0 \\ 1 & 0 & 0 & 3 \end{pmatrix}$$

Gaussian Elimination Example

$$\begin{pmatrix} -24 & 7 & 4 & -6 \\ -8 & 2 & 0 & 0 \\ 5 & 0 & 4 & 0 \\ 1 & 0 & 0 & 3 \end{pmatrix} \quad \begin{pmatrix} -24 & 7 & 4 & -6 \\ 0 & \neq 0 & \neq 0 & \neq 0 \\ 0 & \neq 0 & \neq 0 & \neq 0 \\ 0 & \neq 0 & \neq 0 & \neq 0 \end{pmatrix}$$

$$\begin{pmatrix} -24 & 7 & 4 & -6 \\ -8 & 2 & 0 & 0 \\ 5 & 0 & 4 & 0 \\ 1 & 0 & 0 & 3 \end{pmatrix} \quad \begin{pmatrix} \neq 0 & \neq 0 & \neq 0 & 0 \\ -8 & 2 & 0 & 0 \\ 5 & 0 & 4 & 0 \\ 1 & 0 & 0 & 3 \end{pmatrix}$$

Gaussian Elimination Example

$$\begin{pmatrix} -24 & 7 & 4 & -6 \\ -8 & 2 & 0 & 0 \\ 5 & 0 & 4 & 0 \\ 1 & 0 & 0 & 3 \end{pmatrix} \quad \begin{pmatrix} -24 & 7 & 4 & -6 \\ 0 & \neq 0 & \neq 0 & \neq 0 \\ 0 & \neq 0 & \neq 0 & \neq 0 \\ 0 & \neq 0 & \neq 0 & \neq 0 \end{pmatrix}$$

$$\begin{pmatrix} -24 & 7 & 4 & -6 \\ -8 & 2 & 0 & 0 \\ 5 & 0 & 4 & 0 \\ 1 & 0 & 0 & 3 \end{pmatrix} \quad \begin{pmatrix} \neq 0 & \neq 0 & \neq 0 & 0 \\ -8 & 2 & 0 & 0 \\ 5 & 0 & 4 & 0 \\ 1 & 0 & 0 & 3 \end{pmatrix}$$

- Question: when can we avoid turning zeroes into non-zeroes completely?

Simplification

'Regularity' assumption: *If we add some multiple of row i to row j, at most one non-zero value is turned into a zero.*

Simplification

'Regularity' assumption: *If we add some multiple of row i to row j, at most one non-zero value is turned into a zero.*

- Exact values are not important
- A problem instance is a $n \times n$ $(0, 1)$-matrix M (with m non-zeroes, $n \leq m \leq n^2$):

$$\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{pmatrix}$$

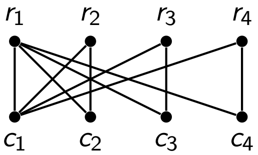
Simplification

'Regularity' assumption: *If we add some multiple of row i to row j, at most one non-zero value is turned into a zero.*

- Exact values are not important
- A problem instance is a $n \times n$ $(0,1)$-matrix M (with m non-zeroes, $n \leq m \leq n^2$):

$$\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{pmatrix}$$

- ...or an equivalent bipartite graph G_M (with m edges):



Suitable Pivots

Remark

A pivot (i, j) (with $M_{i,j} = 1$) does not create additional non-zeroes, if for every i', j' we have that if $M_{i,j'} = 1$ and $M_{i',j} = 1$, then $M_{i',j'} = 1$.

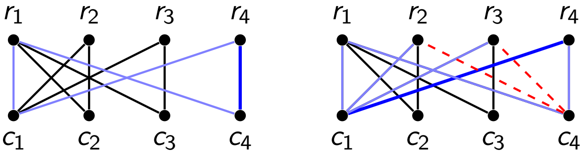
$$\begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}$$

- If we can find a sequence of n such pivots in distinct rows and columns, we can perform elimination without creating new non-zeroes.

Bisimplicial Edges

Definition

An edge e of a bipartite graph G is called *bisimplicial* if the neighbors of the vertices incident to it induce a complete bipartite subgraph.



- Bisimplicial edges in G_M correspond to pivots that avoid new non-zeroes in M.

Perfect Elimination Bipartite Graphs

Definition

(Golumbic and Goss, (1978,1980)) A graph G is called *perfect elimination bipartite* if there exists a sequence of edges $[e_1, e_2, \ldots, e_n]$ such that:

1. e_1 is a bisimplicial edge in G and e_i is bisimplicial in $G - [e_1, \ldots, e_{i-1}]$ for $2 \leq i \leq n$;
2. $G - [e_1, e_2, \ldots, e_n]$ is empty.

- Perfect elimination bipartite graphs correspond to matrices that allow elimination without creating new non-zeroes.
- Naive algorithm for recognition: $\mathcal{O}\left(n^5\right)$

A faster algorithm

Remark

(Goh and Rotem (1982)) Consider the matrix $Q = MM^T$: $Q_{i,j}$ contains the inner product of rows $M_{i,*}$ and $M_{j,*}$. Let l_i equal the number of elements in row $Q_{i,*}$ with value equal to $Q_{i,i}$. Denote by s_j the column sums in M. (i, j) is bisimplicial in G_M iff $M_{i,j} = 1$ and $l_i = s_j$.

- This leads to a $\mathcal{O}\left(n^3\right)$ algorithm
- Spinrad (2004) subsequently improves this to $\mathcal{O}\left(n^3/\log n\right)$

A faster algorithm

Remark

(Goh and Rotem (1982)) Consider the matrix $Q = MM^T$: $Q_{i,j}$ contains the inner product of rows $M_{i,*}$ and $M_{j,*}$. Let l_i equal the number of elements in row $Q_{i,*}$ with value equal to $Q_{i,i}$. Denote by s_j the column sums in M. (i,j) is bisimplicial in G_M iff $M_{i,j} = 1$ and $l_i = s_j$.

- This leads to a $\mathcal{O}(n^3)$ algorithm
- Spinrad (2004) subsequently improves this to $\mathcal{O}(n^3/\log n)$
- Unfortunately, a sparse M may lead to a dense Q:

$$\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} \times \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 4 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix}$$

Summary so far

- Matrices that allow elimination without new non-zeroes correspond to perfect elimination bipartite graphs
- Recognition algorithms (time complexity):
  - naive: $\mathcal{O}\left(n^5\right)$
  - based on matrix-multiplication: $\mathcal{O}\left(n^3/\log n\right)$

Summary so far

- Matrices that allow elimination without new non-zeroes correspond to perfect elimination bipartite graphs
- Recognition algorithms (time complexity):
  - naive: $\mathcal{O}\left(n^5\right)$
  - based on matrix-multiplication: $\mathcal{O}\left(n^3 / \log n\right)$
- However, the result of matrix-multiplication may be a dense matrix, while avoiding new non-zeroes is mainly useful for sparse matrices. . .

An Observation

$$\begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}$$

An Observation

$$\begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix} \qquad \begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}$$

An Observation

$$\begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix} \quad \begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix} \quad \begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}$$

An Observation

$$\begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix} \quad \begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix} \quad \begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}$$

An Observation

$$\begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix} \quad \begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix} \quad \begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix} \quad \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

An Observation

$$\begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix} \quad \begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix} \quad \begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix} \quad \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

An Observation

$$\begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix} \quad \begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix} \quad \begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix} \quad \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix} \quad \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

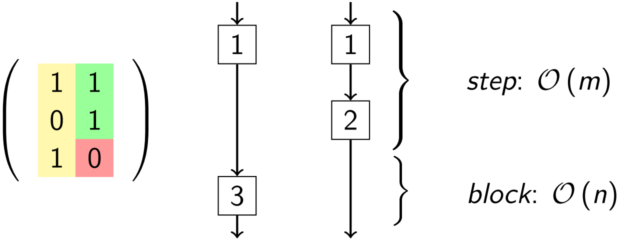
Algorithm Outline

$$\begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix} \quad \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

- Up to *n* iterations (one for each pivot)
- Each iteration, for every edge:
  - Continue checking other edges until blocked
  - If we finish checking other edges, we found a pivot
  - If all edges block, there is no pivot

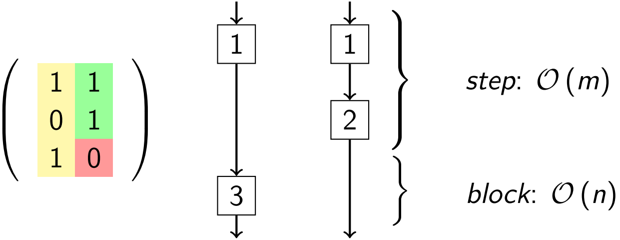
Implementation Details

- Rows/columns of M stored as lists of column/row numbers
- Consider a single edge over the entire algorithm:



Implementation Details

- Rows/columns of M stored as lists of column/row numbers
- Consider a single edge over the entire algorithm:



- Total work for a single edge: $\mathcal{O}(m)$ steps and $\mathcal{O}(n)$ blocks
- For each pivot, we update all list items: $\mathcal{O}(m)$

Time and Space Complexities

- Time complexity: $\mathcal{O}\left(m^2\right)$
  - initialization: $\mathcal{O}\left(n^2\right)$
  - steps: $\mathcal{O}\left(m^2\right)$
  - blocks: $\mathcal{O}\left(nm\right)$
  - updates: $\mathcal{O}\left(nm\right)$
- Space complexity: $\mathcal{O}\left(m\right)$
  - lists: $\mathcal{O}\left(m\right)$
  - edge states: $\mathcal{O}\left(m\right)$
  - row/column data: $\mathcal{O}\left(n\right)$
  - pivots: $\mathcal{O}\left(n\right)$

Conclusion

- Existing literature: focus on time complexity
- However: space complexity is important in practice
- Our new algorithm:
  - $\mathcal{O}\left(m^2\right)$ time
  - $\mathcal{O}\left(m\right)$ space
  - Both time and space improvement for sparse M $\left(m < n\sqrt{n/\log n}\right)$

Conclusion

- Existing literature: focus on time complexity
- However: space complexity is important in practice
- Our new algorithm:
  - $\mathcal{O}\left(m^2\right)$ time
  - $\mathcal{O}\left(m\right)$ space
  - Both time and space improvement for sparse M $\left(m < n\sqrt{n/\log n}\right)$
- Work in progress:
  - Thinking about possible further time complexity improvements
  - Work on alternative elimination procedures