Sparse Linear Systems

Rob H. Bisseling

Mathematical Institute, Utrecht University

Course Introduction Scientific Computing
February 22, 2018

Perfect matching
Block triangular form
Iterative solvers
Neural networks
Summary

Universiteit Utrecht

Outline

Perfect matching

Block triangular form

Iterative solution methods

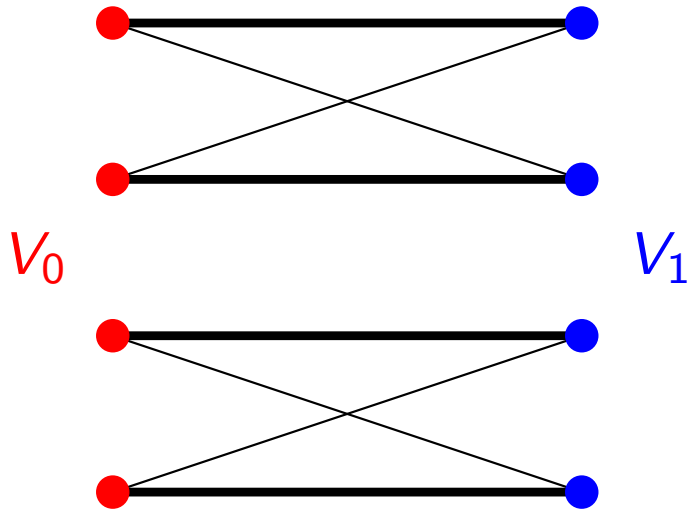
Neural networks

Summary

Perfect matching
Block triangular form
Iterative solvers
Neural networks
Summary

Universiteit Utrecht

A perfect bipartite matching

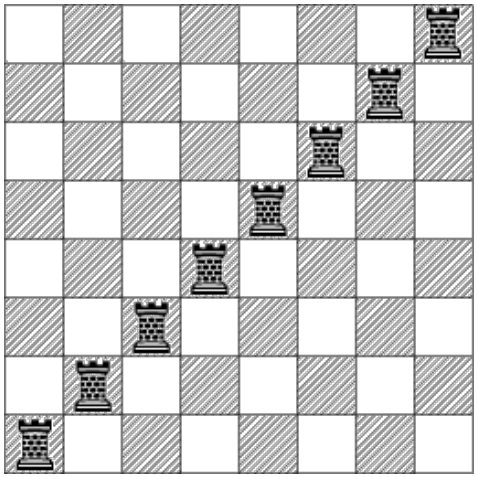


- A bipartite graph has two disjoint sets of vertices: $V = V_0 \cup V_1$ with $E \subseteq V_0 \times V_1$.
- Vertices can represent red/blue, girls/boys, rows/columns.
- A matching in an undirected graph $G = (V, E)$ is a subset $M \subseteq E$ such that $(u, v), (u', v) \in M \implies u = u'$.
- A matching is perfect if all vertices are matched.

Perfect matching
Block triangular form
Iterative solvers
Neural networks
Summary

Universiteit Utrecht

Eight rooks problem



Source: Wolfram MathWorld

- Each rook represents a match between a row and a column.
- No 2 rooks in the same row: a row cannot be matched to 2 columns. Also, no 2 rooks in the same column.
- 8 rooks form a perfect matching $M = \{a1, b2, c3, d4, e5, f6, g7, h8\}$.
- $|M| = 8$, $|V| = 16$ and $|E| = 64$.
- Problem is harder with queens!

Perfect matching
Block triangular form
Iterative solvers
Neural networks
Summary

Universiteit Utrecht

Chemical process problem b1_ss

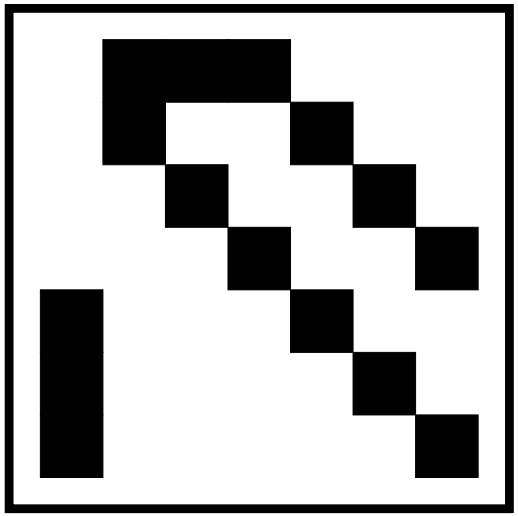
$$\begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0.45 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0.1 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0.45 \\ -0.03599942 & 0 & 0 & 0 & 1 & 0 & 0 \\ -0.0176371 & 0 & 0 & 0 & 0 & 1 & 0 \\ -0.007721779 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \mathbf{x} = \begin{bmatrix} -0.0001 \\ 0.1167 \\ -0.2333 \\ 0.1167 \\ -0.4993128 \\ 0.3435885 \\ 0.7467878 \end{bmatrix}$$

Perfect matching
Block triangular form
Iterative solvers
Neural networks
Summary

- Solving $A\mathbf{x} = \mathbf{b}$ for unsymmetric 7×7 matrix A with 15 nonzeros from chemical process simulation.
- Variation in numerical values is sometimes caused by choice of physical units. Scaling A into $A' = D_0 A D_1$ with D_0, D_1 diagonal matrices may help.

Universiteit Utrecht

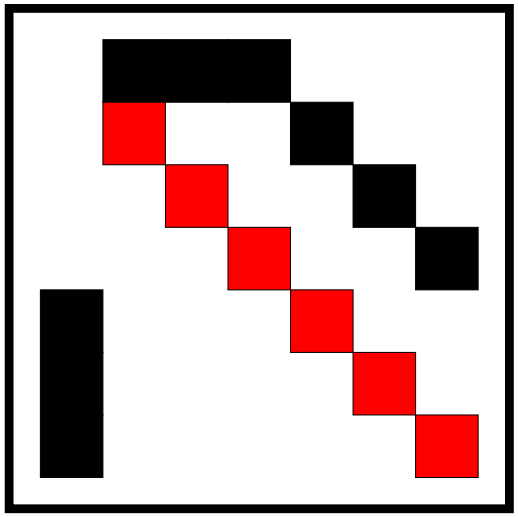
Choosing pivots in matrix b1_ss



- Choose matches from the nonzeros, placing rooks on the chessboard on locations allowed by sparsity pattern.

Perfect matching
Block triangular form
Iterative solvers
Neural networks
Summary

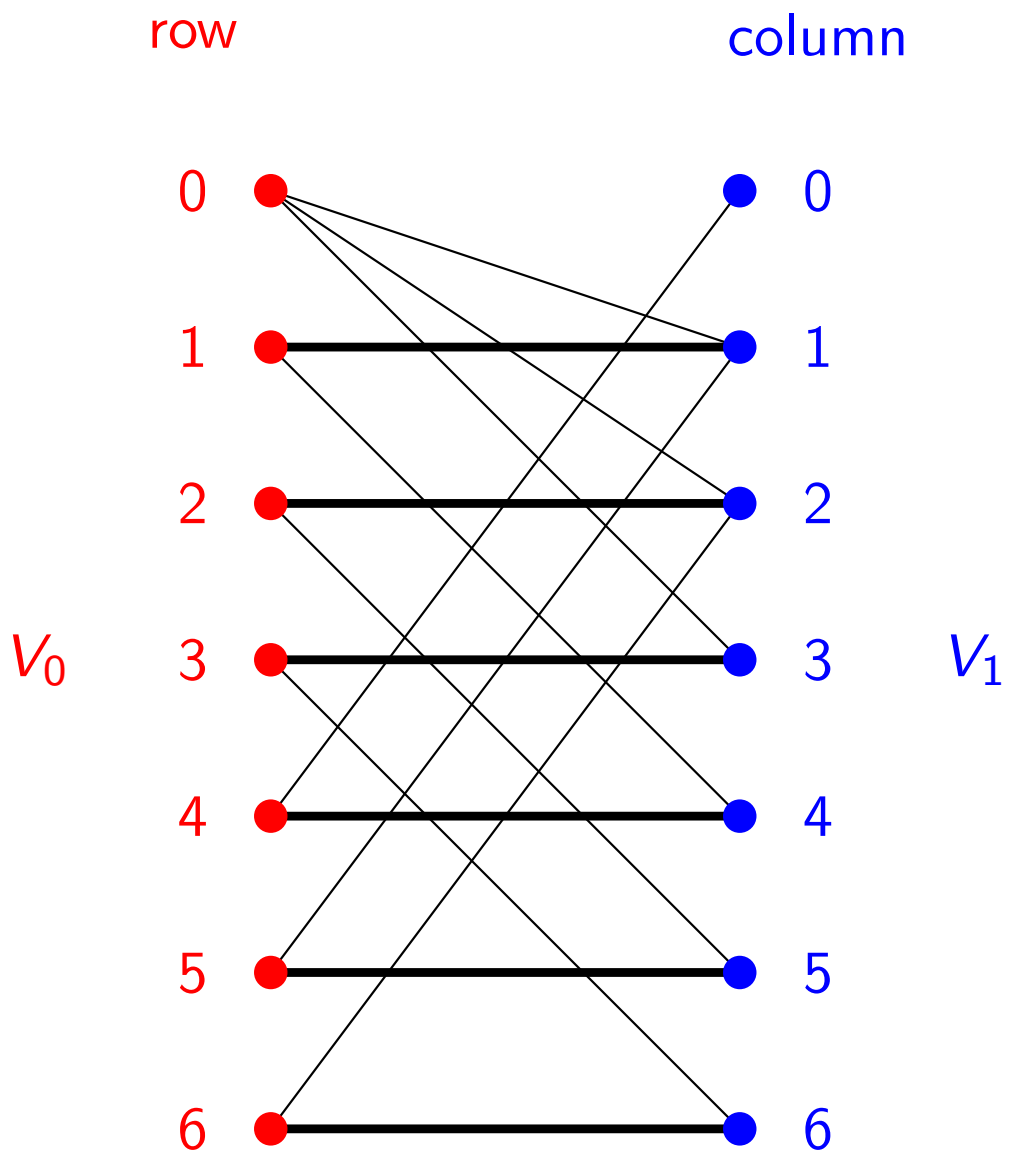
First choice: 6 pivots



- Matching with 6 matches, shown in red.
- All 6 pivots have $|a_{ij}| = 1$, which is the maximum numerical absolute value, good for stability.
- Can we find a perfect matching, with 7 pivots?

Perfect matching
Block triangular form
Iterative solvers
Neural networks
Summary

Matching in bipartite graph of matrix b1_ss



Perfect matching
Block triangular form
Iterative solvers
Neural networks
Summary

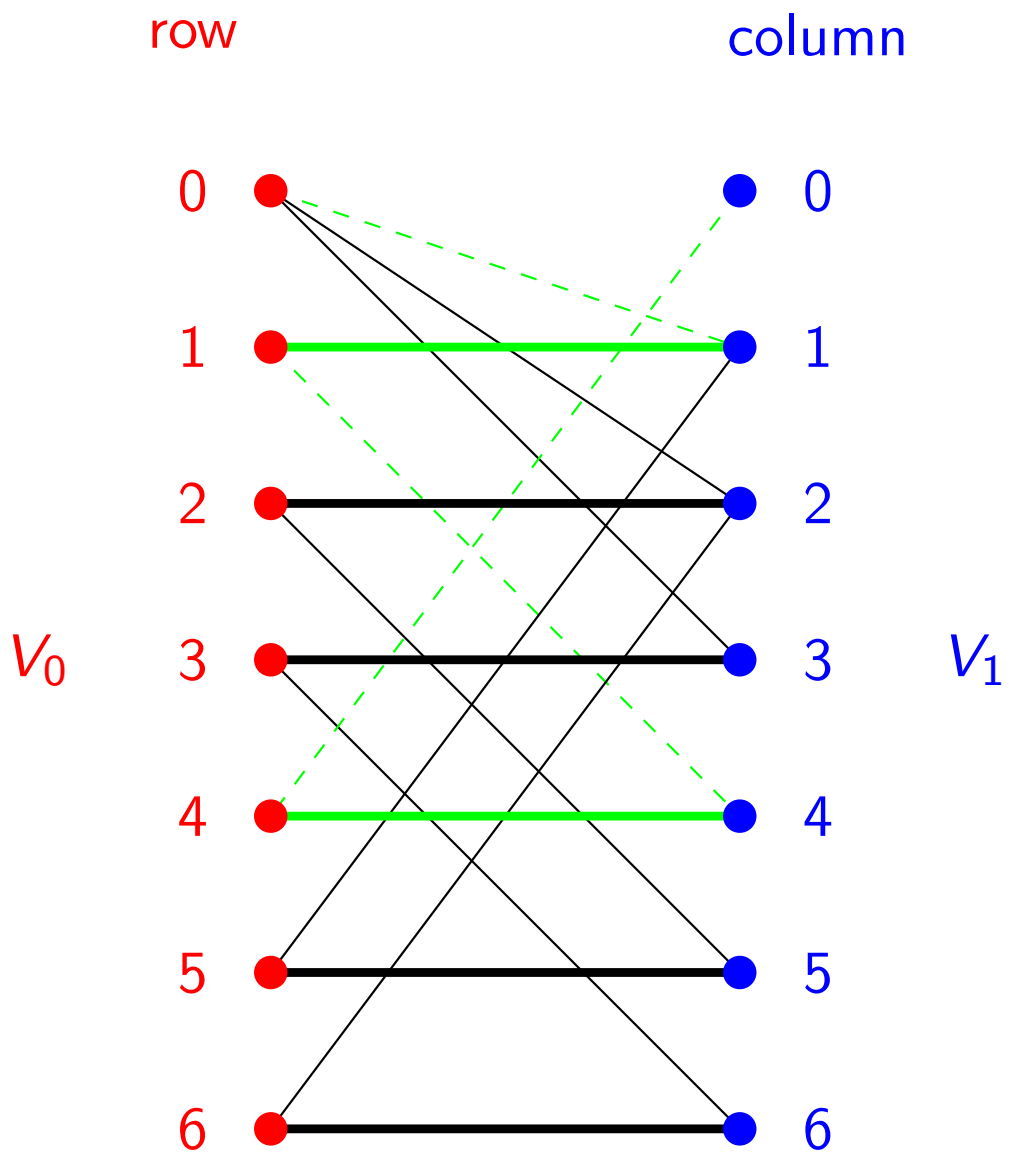
Universiteit Utrecht

Augmenting path

- An alternating path in a graph is a path $v_0, v_1, \ldots, v_n$ in V with $(v_i, v_{i+1}) \in E$ for all $i = 0, \ldots, n-1$, where edges are alternatingly in M and not in M.
- An augmenting path is an alternating path with $(v_0, v_1), (v_{n-1}, v_n) \notin M$. (Thus the path must have odd length.)
- We can flip the edges of an augmenting path and obtain an extra match.
- Theorem (Berge 1957): a matching has maximum cardinality if and only if there exists no augmenting path.

Perfect matching
Block triangular form
Iterative solvers
Neural networks
Summary

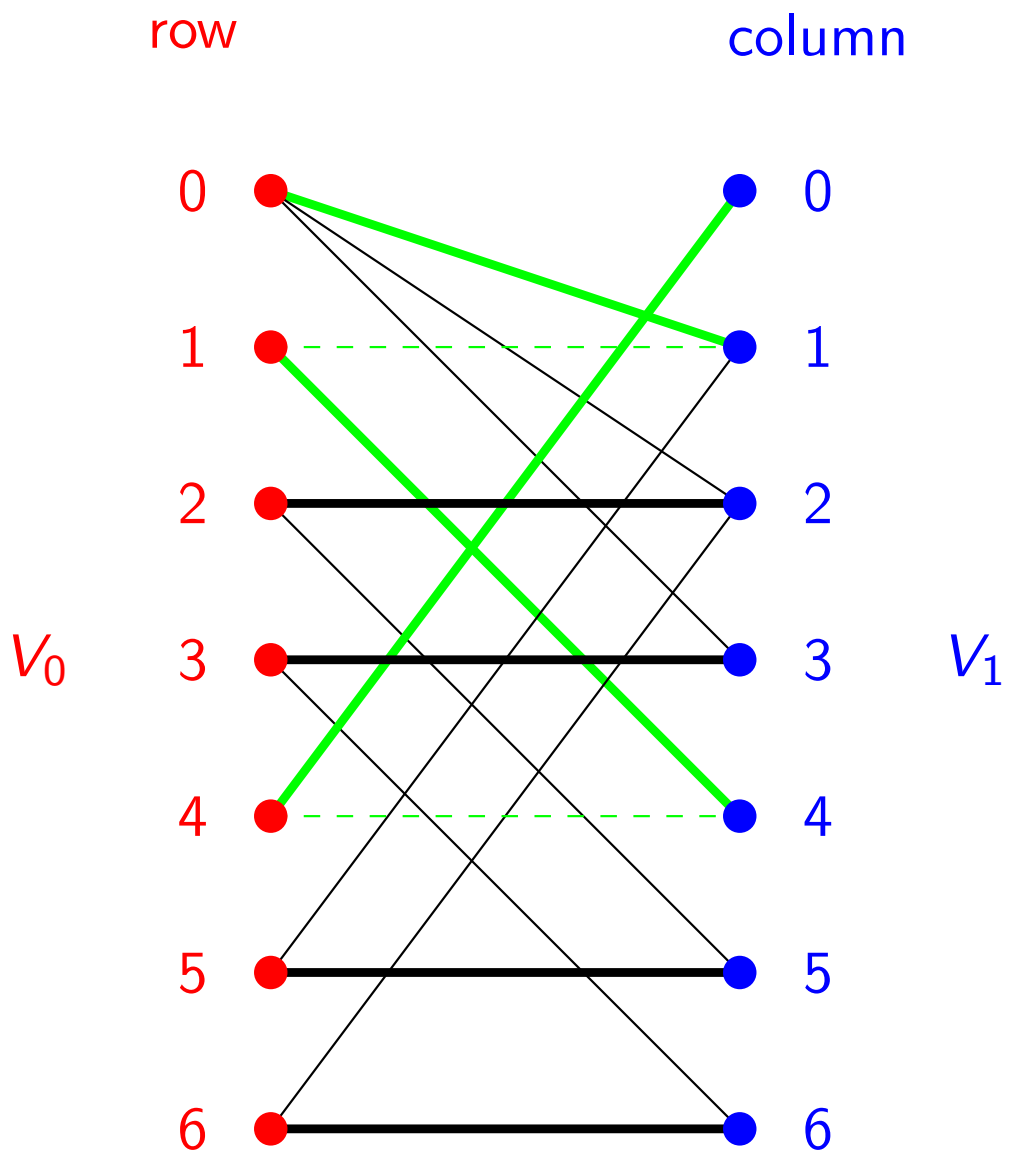
Augmenting path (in green) in bipartite graph



Perfect matching
Block triangular form
Iterative solvers
Neural networks
Summary

Universiteit Utrecht

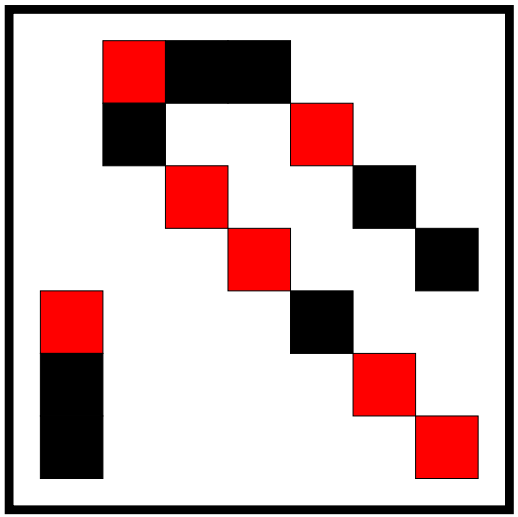
Flip augmenting path



Perfect matching
Block triangular form
Iterative solvers
Neural networks
Summary

Universiteit Utrecht

Perfect matching: 7 pivots



Perfect matching
Block triangular form
Iterative solvers
Neural networks
Summary

- Diagonal strength is

$$\begin{aligned} |a_{40}a_{01}a_{22}a_{33}a_{14}a_{55}a_{66}| &= |-0.03599942 \cdot 1 \cdot -1 \cdot -1 \cdot 0.45 \cdot 1 \cdot 1| \\ &\approx 0.0162. \end{aligned}$$

- This strength cannot be improved. (Proof: look at numerical values of column 0).

Universiteit Utrecht

Life is a compromise

- Maintain sparsity by Markowitz criterion: choose nonzero a_{ij} with minimal upper bound $(r_i - 1)(c_j - 1)$ on the number of fill-ins.
- Maintain numerical stability: choose largest absolute value $|a_{ij}|$.
- Compromise: choose candidate with lowest Markowitz count that satisfies

$$|a_{ij}| \geq u \cdot \max_r |a_{rj}|$$

where the threshold parameter u is often chosen as $u = 0.1$.

Perfect matching
Block triangular form
Iterative solvers
Neural networks
Summary

Maximum matching algorithm

- The Hopcroft–Karp (1973) algorithm has a time complexity of $\mathcal{O}(\sqrt{|V|} \cdot |E|)$.
- It computes a Breadth-First Search (BFS) tree, where each path from the root (an unmatched vertex) to a leaf in the tree is an alternating path in the graph.
- The algorithm repeatedly searches for a maximal set of vertex-disjoint shortest augmenting paths.
- It can be initialised with a heuristic algorithm, which speeds up the search. E.g. Karp–Sipser first matches vertices with degree 1. If no such vertex exists, it randomly matches a vertex.

Perfect matching
Block triangular form
Iterative solvers
Neural networks
Summary

Strengthening the diagonal (Duff and Koster 1999)

- If the maximum matching is perfect (all vertices are matched), we have positive diagonal strength.
- We want to maximise this strength, equivalent to finding a permutation π of the matrix columns that maximises

$$\prod_{i=0}^{n-1} |a_{i\pi(i)}|.$$

- Implementation in the successful code MC64 by Iain Duff and Jacko Koster in the Harwell Software Library.

Perfect matching
Block triangular form
Iterative solvers
Neural networks
Summary

Connection to classic assignment problem

- We want to maximise $\prod_{i=0}^{n-1} |a_{i\pi(i)}|$.
- Define $a_j = \max_i |a_{ij}|$, the maximum absolute value in column j. Thus, we want to minimise

$$\log \frac{\prod_{i=0}^{n-1} a_{\pi(i)}}{\prod_{i=0}^{n-1} |a_{i\pi(i)}|} = \sum_{i=0}^{n-1} (\log a_{\pi(i)} - \log |a_{i\pi(i)}|) = \sum_{i=0}^{n-1} c_{i\pi(i)},$$

  where we define $c_{ij} = \log a_j - \log |a_{ij}|$ if $a_{ij} \neq 0$, and $c_{ij} = \infty$ otherwise.
- Here, $c_{ij} > 0$ can be seen the cost of assigning task i to worker j in the linear sum assignment problem.
- The goal is then to find a perfect matching M with minimum weight $c(M)$.

Perfect matching
Block triangular form
Iterative solvers
Neural networks
Summary

Upper block triangular form

- Let A be a square, nonsingular matrix,

$$A = \begin{bmatrix} A_{00} & A_{01} & \cdots & A_{0,k-1} \\ 0 & A_{11} & \cdots & A_{1,k-1} \\ \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & A_{k-1,k-1} \end{bmatrix},$$

  where each block A_{ii} is square and nonsingular.
- Then we say that A has upper block triangular form (upper BTF) with k blocks.
- A matrix A is irreducible if the only upper BTF we can obtain by permuting its rows and columns has $k = 1$.

Perfect matching
Block triangular form
Iterative solvers
Neural networks
Summary

Using the upper block triangular form

- Let A be a matrix in upper BTF,

$$A = \begin{bmatrix} A_{00} & A_{01} & \cdots & A_{0,k-1} \\ 0 & A_{11} & \cdots & A_{1,k-1} \\ \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & A_{k-1,k-1} \end{bmatrix}.$$

- If we write $\mathbf{x} = [\mathbf{x}_0, \ldots, \mathbf{x}_{k-1}]^T$, and similarly for $\mathbf{b}$, we can rewrite the system $A\mathbf{x} = \mathbf{b}$ as

$$A_{ii}\mathbf{x}_i = \mathbf{b}_i - \sum_{j=i+1}^{k-1} A_{ij}\mathbf{x}_j, \quad \text{for } j = k-1, \ldots, 0.$$

  and hence solve the system by block backward substitution.
- The linear system solution reduces to k smaller linear system solutions, probably with less fill-in, and $\frac{k(k-1)}{2}$ matrix–vector multiplications (without fill-in!).

Perfect matching
Block triangular form
Iterative solvers
Neural networks
Summary

Obtaining an upper block triangular form

1. Obtain a perfect matching and permute A into $A' = AQ$ with a zero-free diagonal ($a'_{ii} \neq 0$ for all i).
2. Construct the directed graph corresponding to A' by: V is the set of rows/columns, $|V| = n$, and there is an edge from i to j, i.e., $(i, j) \in E$ if and only if $a'_{ij} \neq 0$ and $i \neq j$.
3. Obtain strongly connected components, i.e., groups of vertices that can all reach each other.

Perfect matching
Block triangular form
Iterative solvers
Neural networks
Summary

Obtaining strongly connected components

1. Start in a vertex v_0 and follow a path along the directed edges.
2. If the path ends in a cycle, all the vertices of the cycle belong to the same strongly connected component.
3. Then merge the vertices or mark them using a stack data structure (Tarjan 1972) as belonging to the same block.
4. If the path does not end in a cycle, it terminates at a dead end, a vertex with no outgoing edges, corresponding to an empty row in A' (except for the diagonal element). Renumber this vertex as the last.
5. Continue in the same way starting at the previous vertex on the path or an arbitrary remaining vertex.

Perfect matching
Block triangular form
Iterative solvers
Neural networks
Summary

Complexity of Tarjan's algorithm

- The devil is in the implementation details.
- If this is done right, the time complexity is $\mathcal{O}(|V| + |E|) = \mathcal{O}(n + nz)$.

Perfect matching
Block triangular form
Iterative solvers
Neural networks
Summary

Dulmage-Mendelsohn decomposition

In Matlab:

$$[\mathtt{p}, \mathtt{q}, \mathtt{r}, \mathtt{s}] = \mathtt{dmperm}(\mathtt{A}),$$

where `p` and `q` are permutations such that `A(p,q)` is block upper triangular. The block sizes are given by `r` and `s`.

Perfect matching
Block triangular form
Iterative solvers
Neural networks
Summary

Iterative solution methods

- Solve $A\mathbf{x} = \mathbf{b}$ by starting with an initial guess $\mathbf{x}^{(0)}$ and successively getting better guesses $\mathbf{x}^{(i)}$ in each iteration i, until convergence $||A\mathbf{x}^{(i)} - \mathbf{b}|| < \epsilon$.
- Algorithms: conjugate gradients (CG) for symmetric systems, generalised minimal residual (GMRES), BiCGStab, IDR(s) for unsymmetric systems.
- They involve A in sparse matrix–vector multiplication, multiplying by A and/or A^T, and they also perform some linear vector operations.
- The matrix does not change. No fill-in!
- The ordering of rows and columns is still important.

Perfect matching
Block triangular form
Iterative solvers
Neural networks
Summary

Preconditioning

- A preconditioner M is an $n \times n$ matrix such that $MA\mathbf{x} = M\mathbf{b}$ is easier to solve than the original system $A\mathbf{x} = \mathbf{b}$.
- This works perfectly if $M = A^{-1}$ (alas, hard to obtain!), but it may still work well if $M \approx A^{-1}$.
- The system $M\mathbf{y} = \mathbf{c}$ must be easy to solve. For example, M is a diagonal matrix (Jacobi preconditioner), or M is very sparse.

Perfect matching
Block triangular form
Iterative solvers
Neural networks
Summary

Incomplete factorisation

- Incomplete Cholesky factorisation IC(0), proposed by van der Vorst and Meijerink (1977): allow no fill-in. Just compute with the original sparsity pattern of A.
- Incomplete LU decomposition ILU(0): same for LU in case of an unsymmetric matrix A.
- $M = ILU(0)$ is often a good preconditioner for A.
- In general, we can allow the creation of new nonzeros either based on the location in the matrix, or the numerical value (ILU(τ) with τ a threshold value), or a mix.

Perfect matching
Block triangular form
Iterative solvers
Neural networks
Summary

Allowing limited fill-in

- The basic update operation in stage k of LU decomposition is

$$a_{ij}^{(k+1)} := a_{ij}^{(k)} - \frac{a_{ik}^{(k)} a_{kj}^{(k)}}{a_{kk}^{(k)}}.$$

- For the newborn baby $a_{ij}^{(k+1)}$, the father is $a_{ik}^{(k)}$, and the mother $a_{kj}^{(k)}$.
- Initialise the fill-in level of a matrix element by

$$lev_{ij} = \begin{cases} 0 & \text{if } a_{ij} \neq 0 \text{ or } i = j \\ \infty & \text{otherwise.} \end{cases}$$

- Update the fill-in level in a run of the ILU algorithm by

$$lev_{ij} := \min(lev_{ij}, lev_{ik} + lev_{kj} + 1).$$

Perfect matching
Block triangular form
Iterative solvers
Neural networks
Summary

Allowing limited fill-in

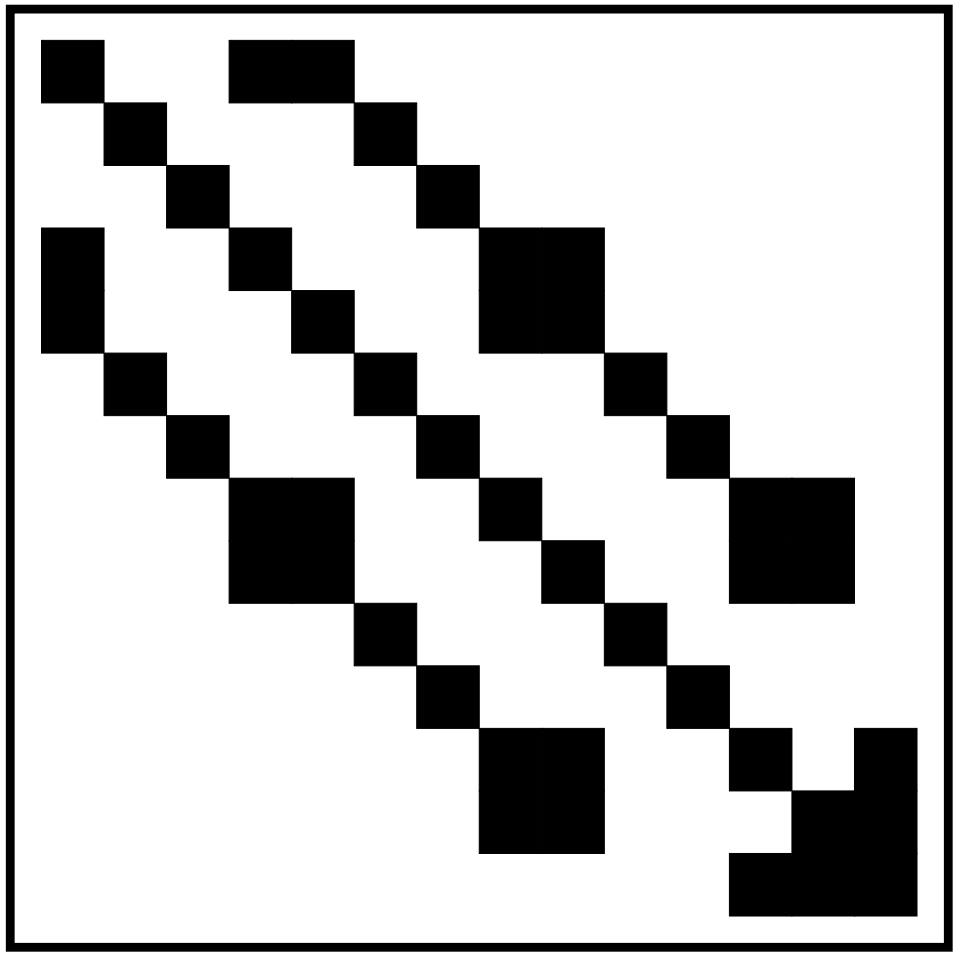
- Because of the update

$$lev_{ij} := \min(lev_{ij}, lev_{ik} + lev_{kj} + 1),$$

  the level of an element decreases or stays the same during the ILU algorithm.
- Motivation: think of values becoming smaller when created by fill-in. For example, $a_{ij} = 2^{-lev_{ij}}$.

Perfect matching
Block triangular form
Iterative solvers
Neural networks
Summary

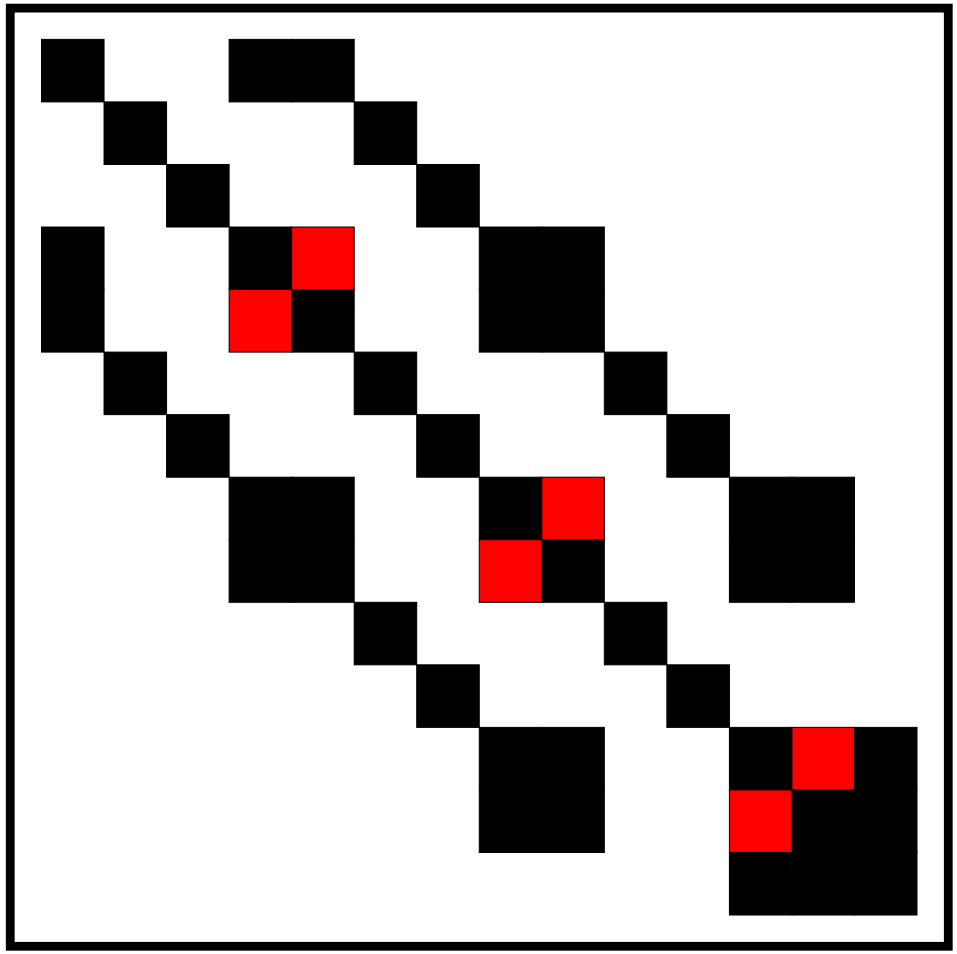
ILU(0) for matrix LFAT5



Perfect matching
Block triangular form
Iterative solvers
Neural networks
Summary

Universiteit Utrecht

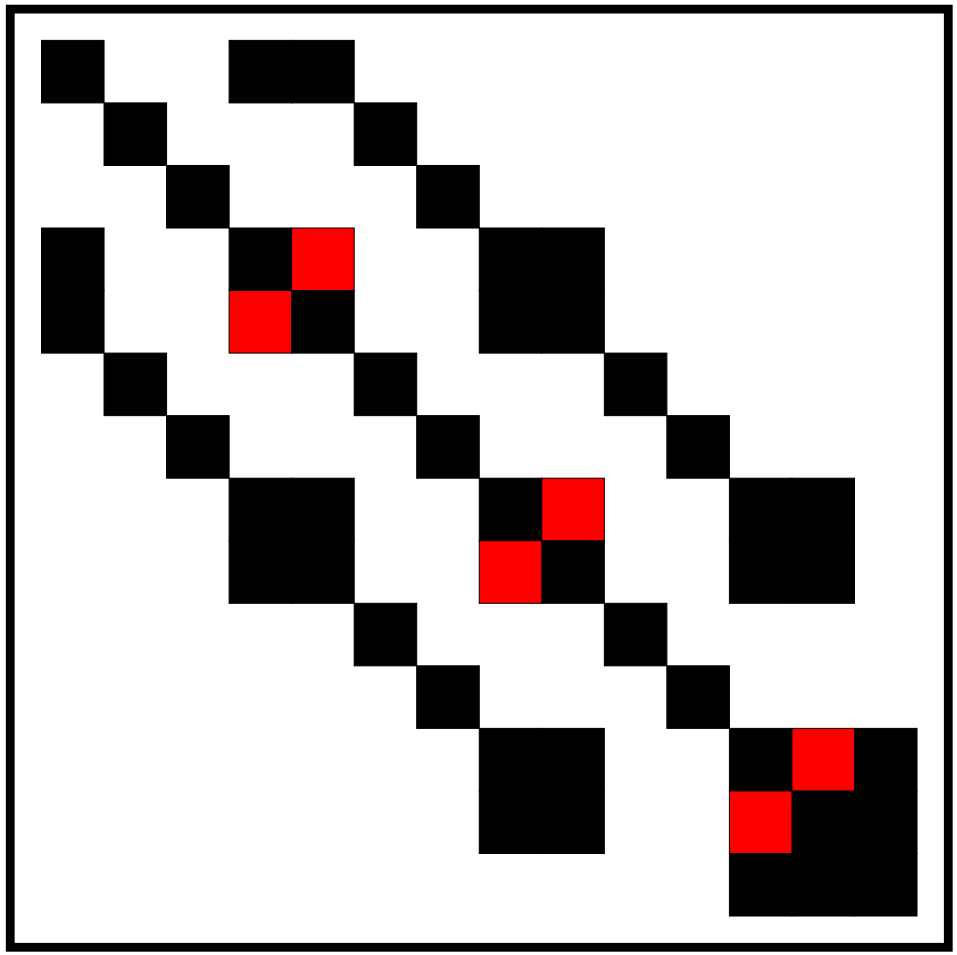
ILU(1) for matrix LFAT5



Perfect matching
Block triangular form
Iterative solvers
Neural networks
Summary

Universiteit Utrecht

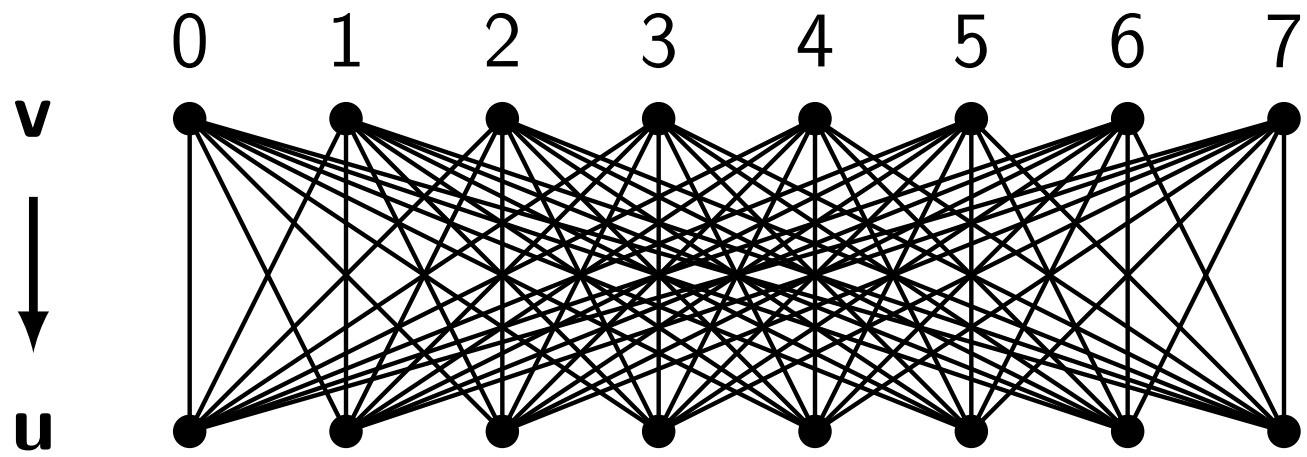
No further fill-in for matrix LFAT5



Perfect matching
Block triangular form
Iterative solvers
Neural networks
Summary

Universiteit Utrecht

Two layers of an artificial neural network



- The strength v_j of a signal fired by top neuron j is an input to the strength u_i of bottom neuron i.
- The connection between j and i carries a weight a_{ij} determined during a learning phase of the network.
- A weighted average is computed and then a nonlinear activation function such as $f(x) = \frac{1}{1+e^{-x}}$ is applied:

$$u_i = f\left(\sum_{j=0}^{n-1} a_{ij} v_j\right).$$

Perfect matching
Block triangular form
Iterative solvers
Neural networks
Summary

Summary

- We have formulated the process of finding n pivots in an $n \times n$ matrix A as a bipartite matching problem.
- A perfect matching leads to a zero-free diagonal.
- A strong diagonal can be obtained by using matching variants (Duff and Koster) or by solving a minimum-weight assignment problem.
- The block triangular form for unsymmetric matrices can save a lot of work. It can be found using strongly connected components in a directed graph.
- Incomplete factorisation can be used as a preconditioner to speed up iterative solvers.
- Using a sparse artificial neural network is almost the same as multiplying a sparse matrix and a vector.

Perfect matching
Block triangular form
Iterative solvers
Neural networks
Summary