Modeling Physics with Differential-Algebraic Equations

Lecture 2

Structural Analysis: Index Reduction

COMASIC (M2)
December 13th, 2017

Khalil Ghorbal
khalil.ghorbal@inria.fr

Summary lecture 1

1. Ordinary Differential Equations:
   - Cauchy-Lipschitz theorem: existence and uniqueness of solutions
   - Liouville theorem: no closed form solutions in general
   - Numerical integration: convergence and stability
   - Qualitative analysis: invariant regions

2. Differential-Algebraic Equations
   - Informal Introduction
   - Examples
   - Different Forms

Outline

1. Matching Problem
2. BLT Decomposition
3. Pantelides Algorithm

Structural Analysis of Systems of Equations: Example

Consider the following system of equations:

$$\begin{cases} f_1(x_1) = 0 \\ f_2(x_1, x_2, x_3) = 0 \\ f_3(x_3) = 0 \end{cases}$$

Structural Analysis of Systems of Equations: Example

Consider the following system of equations:

$$\begin{cases} f_1(x_1) = 0 \\ f_2(x_1, x_2, x_3) = 0 \\ f_3(x_3) = 0 \end{cases}$$

Incidence Matrix

$$\begin{array}{cc} & \begin{array}{ccc} x_1 & x_2 & x_3 \end{array} \\ \begin{array}{c} f_1 \\ f_2 \\ f_3 \end{array} & \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix} \end{array}$$

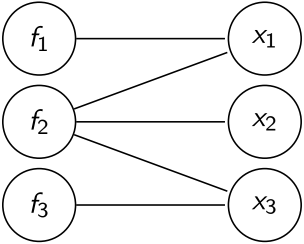
Consider the following system of equations:

$$\begin{cases} f_1(x_1) = 0 \\ f_2(x_1, x_2, x_3) = 0 \\ f_3(x_3) = 0 \end{cases}$$

Incidence Matrix

$$\begin{array}{cc} & \begin{array}{ccc} x_1 & x_2 & x_3 \end{array} \\ \begin{array}{c} f_1 \\ f_2 \\ f_3 \end{array} & \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix} \end{array}$$

Bipartite Graph (Bigraph)



Bipartite graph (F, V, E)

- F: set of equations
- V: set of variables (disjoint with F)
- E: subset of the cartesian product $F \times V$

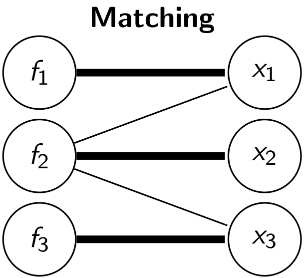
Example: a triangle is **not** a bipartite graph.

Matching Problem

Given a bipartite graph (F, V, E), assign one and only one equation $f \in F$ to each variable $v \in V$ such that $(f, v) \in E$.

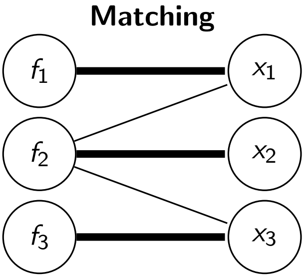
Algorithm

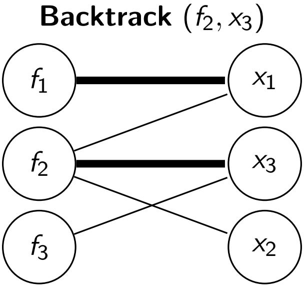
For each equation, pick up its first **unmatched** variable. The procedure succeeds whenever all variables are matched.



Algorithm **(incomplete/wrong)**

For each equation, pick up its first **unmatched** variable. The procedure succeeds whenever all variables are matched.





Let $G : (V, E)$ denote a graph

- **Matching**: A matching is a set of pairwise non-adjacent edges.
- **Maximum Matching**: A matching having the maximum cardinality, denoted $\nu(G)$.
- **Perfect Matching**: A matching such that every vertex of the graph is incident to exactly one edge of the matching.

Let $G : (V, E)$ denote a graph

- **Matching**: A matching is a set of pairwise non-adjacent edges.
- **Maximum Matching**: A matching having the maximum cardinality, denoted $\nu(G)$.
- **Perfect Matching**: A matching such that every vertex of the graph is incident to exactly one edge of the matching.

Let $G : (V, E)$ denote a graph

- **Matching**: A matching is a set of pairwise non-adjacent edges.
- **Maximum Matching**: A matching having the maximum cardinality, denoted $\nu(G)$.
- **Perfect Matching**: A matching such that every vertex of the graph is incident to exactly one edge of the matching.

Matching Problem: Algorithm

Let $G : (V, E)$ denote a graph

- **Matching**: A matching is a set of pairwise non-adjacent edges.
- **Maximum Matching**: A matching having the maximum cardinality, denoted $\nu(G)$.
- **Perfect Matching**: A matching such that every vertex of the graph is incident to exactly one edge of the matching.

Proposition G has a perfect matching if and only if $|G| = 2\nu(G)$.

⊶ For systems of n variables and n equations: (1) compute $\nu(G)$, (2) check if $\nu(G) = n$.

Hopcroft-Karp Algorithm (1973)

- Input: bipartite graph (U, V, E)
- Output: maximum cardinality matching
- Complexity: (worst case) $O(|E|\sqrt{|V|})$

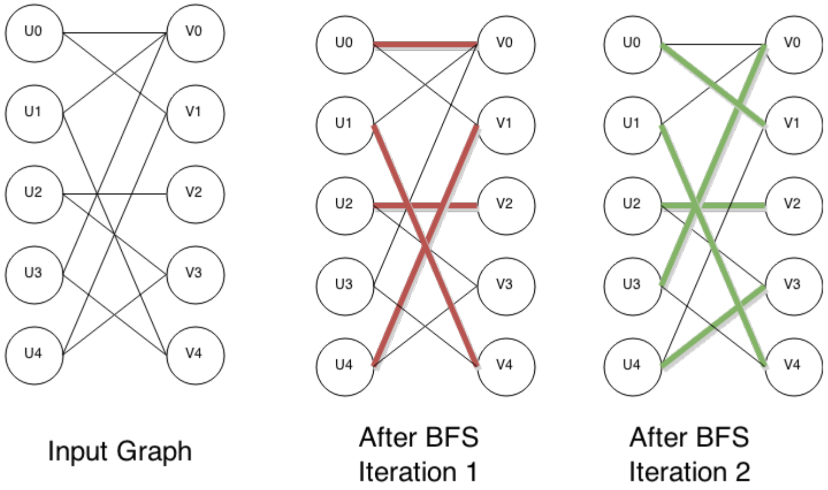
Augmented Shortest Path

At each phase:

- BFS: alternate between U and V where one starts from an unmatched variable in U and reaches an unmatched variable in V while following a matched edge from V to U (this gives an augmented shortest path).
- DFS: selects one shortest path out of the many selected ones by the BFS.
- update the matching set

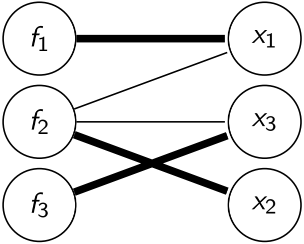
Example

Source: Wikipedia



Maximum Transversal Problem

Matching



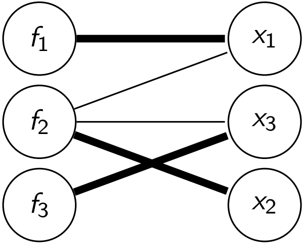
Incidence Matrix

$$\begin{array}{c} \\ f_1 \\ f_2 \\ f_3 \end{array} \begin{array}{c} \begin{array}{ccc} x_1 & x_2 & x_3 \end{array} \\ \begin{pmatrix} \boxed{1} & 0 & 0 \\ 1 & \boxed{1} & 1 \\ 0 & 0 & \boxed{1} \end{pmatrix} \end{array}$$

Maximum Transversal Problem

Finding a permutation that places a maximum number of non-zero on the diagonal of a sparse matrix.

Matching



Incidence Matrix

$$\begin{array}{c c} & \begin{array}{ccc} x_1 & x_2 & x_3 \end{array} \\ \begin{array}{c} f_1 \\ f_2 \\ f_3 \end{array} & \begin{pmatrix} \boxed{1} & 0 & 0 \\ 1 & \boxed{1} & 1 \\ 0 & 0 & \boxed{1} \end{pmatrix} \end{array}$$

$$\begin{array}{c c} & \begin{array}{ccc} x_1 & x_3 & x_2 \end{array} \\ \begin{array}{c} f_1 \\ f_3 \\ f_2 \end{array} & \begin{pmatrix} \boxed{1} & - & - \\ 0 & \boxed{1} & - \\ 1 & 1 & \boxed{1} \end{pmatrix} \end{array}$$

Maximum Transversal Problem

Finding a permutation that places a maximum number of non-zero on the diagonal of a sparse matrix.

- Very useful prior to decomposing the matrix using Gaussian elimination.

Goal: Given a square matrix M ($m_{ij} \in \{0, 1\}$), find a permutation to put M in a block lower triangular form.

$$\begin{array}{c} \\ f_1 \\ f_2 \\ f_3 \\ f_4 \end{array} \begin{array}{c} \begin{array}{cccc} x_1 & x_2 & x_3 & x_4 \end{array} \\ \begin{pmatrix} 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \end{pmatrix} \end{array}$$

$$\begin{array}{c} \\ f_3 \\ f_1 \\ f_4 \\ f_2 \end{array} \begin{array}{c} \begin{array}{cccc} x_2 & x_1 & x_3 & x_4 \end{array} \\ \begin{pmatrix} \mathbf{1} & 0 & 0 & 0 \\ 0 & \mathbf{1} & \mathbf{1} & 0 \\ 1 & \mathbf{1} & \mathbf{1} & 0 \\ 1 & 1 & 0 & \mathbf{1} \end{pmatrix} \end{array}$$

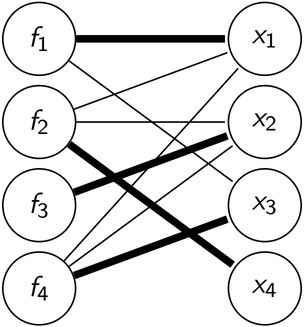
Blocks of dimension > 1 (called **algebraic loops**) are (numerically) solved by Gaussian elimination if linear or Newton methods if nonlinear.

Blocks of dimension > 1 (called **algebraic loops**) are (numerically) solved by Gaussian elimination if linear or Newton methods if nonlinear.

Three main steps

1. Find a (perfect) matching
2. Construct a *dependency graph*
3. Find *strongly connected components* (Tarjan's algorithm)

Bipartite Graph (Bigraph)



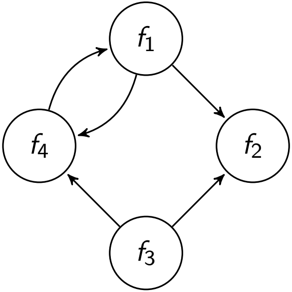
Incidence Matrix

$$\begin{array}{c c} & \begin{array}{cccc} x_1 & x_2 & x_3 & x_4 \end{array} \\ \begin{array}{c} f_1 \\ f_2 \\ f_3 \\ f_4 \end{array} & \begin{pmatrix} \boxed{1} & 0 & 1 & 0 \\ 1 & 1 & 0 & \boxed{1} \\ 0 & \boxed{1} & 0 & 0 \\ 1 & 1 & \boxed{1} & 0 \end{pmatrix} \end{array}$$

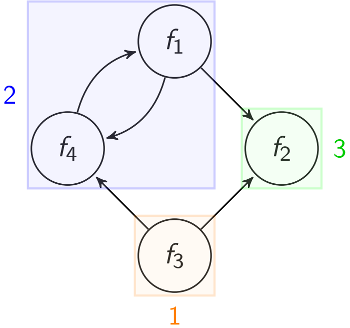
Incidence Matrix

$$
\begin{array}{c c}
 & \begin{array}{cccc} x_1 & x_2 & x_3 & x_4 \end{array} \\
\begin{array}{c} f_1 \\ f_2 \\ f_3 \\ f_4 \end{array} &
\begin{pmatrix}
\boxed{1} & 0 & 1 & 0 \\
1 & 1 & 0 & \boxed{1} \\
0 & \boxed{1} & 0 & 0 \\
1 & 1 & \boxed{1} & 0
\end{pmatrix}
\end{array}
$$

Dependency Graph



Dependency Graph



BLT Decomposition

$$\begin{array}{c} \\ f_3 \\ f_1 \\ f_4 \\ f_2 \end{array} \begin{array}{c} \begin{array}{cccc} x_2 & x_1 & x_3 & x_4 \end{array} \\ \begin{pmatrix} \mathbf{1} & 0 & 0 & 0 \\ 0 & \mathbf{1} & \mathbf{1} & 0 \\ 1 & \mathbf{1} & \mathbf{1} & 0 \\ 1 & 1 & 0 & \mathbf{1} \end{pmatrix} \end{array}$$

Definitions

- A **directed** graph is strongly connected if every vertex is reachable from every other vertex.
- **Strongly connected components** of a directed graph form a partition into subgraphs which are themselves strongly connected.
- By contracting each strongly connected component into one vertex, one obtains a **condensation** of the original graph into a **directed acyclic graph**.

Definitions

- A **directed** graph is strongly connected if every vertex is reachable from every other vertex.
- **Strongly connected components** of a directed graph form a partition into subgraphs which are themselves strongly connected.
- By contracting each strongly connected component into one vertex, one obtains a **condensation** of the original graph into a **directed acyclic graph**.

Definitions

- A **directed** graph is strongly connected if every vertex is reachable from every other vertex.
- **Strongly connected components** of a directed graph form a partition into subgraphs which are themselves strongly connected.
- By contracting each strongly connected component into one vertex, one obtains a **condensation** of the original graph into a **directed acyclic graph**.

Definitions

- A **directed** graph is strongly connected if every vertex is reachable from every other vertex.
- **Strongly connected components** of a directed graph form a partition into subgraphs which are themselves strongly connected.
- By contracting each strongly connected component into one vertex, one obtains a **condensation** of the original graph into a **directed acyclic graph**.

Tarjan's Algorithm (1972)

- One depth-first search (vs. 2 for Kosaraju's algorithm (1978))
- Simple and elegant data structure
- Complexity: $O(|V| + |E|)$

General Form

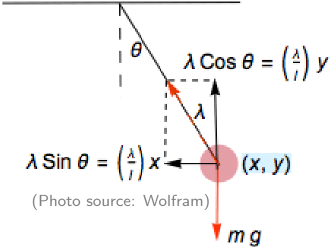
$$F(x, \dot{x}, y, t) = 0$$

- state variables: $x \in \mathbb{R}^n$
- algebraic variables: $y \in \mathbb{R}^m$
- $F : G \subset \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R} \to \mathbb{R}^{m+n}$

Index

The index of an DAE is the minimum number of times that all or part of the DAE must be differentiated with respect to time (t) in order to determine $\dot{x}$ as a continuous function of x and t.

(Brenan, Campbell 1996)



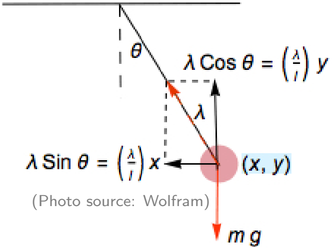
(Photo source: Wolfram)

System of Equations

$$
\begin{aligned}
f_1 :&\quad \dot{x} &= u \\
f_2 :&\quad \dot{y} &= v \\
f_3 :&\quad \dot{u} &= -\lambda x \\
f_4 :&\quad \dot{v} &= -\lambda y - g \\
f_5 :&\quad 0 &= L^2 - x^2 - y^2
\end{aligned}
$$

Incidence Matrix

$$
\begin{array}{c|c}
 & \begin{matrix} \dot{x} & \dot{y} & \dot{u} & \dot{v} & \lambda \end{matrix} \\
\begin{matrix} f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \end{matrix} & \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}
\end{array}
$$



(Photo source: Wolfram)

System of Equations

$$
\begin{aligned}
f_1 : \quad & \dot{x} && = u \\
f_2 : \quad & \dot{y} && = v \\
f_3 : \quad & \dot{u} && = -\lambda x \\
f_4 : \quad & \dot{v} && = -\lambda y - g \\
f_5 : \quad & 0 && = L^2 - x^2 - y^2
\end{aligned}
$$

Incidence Matrix

$$
\begin{array}{c c}
 & \begin{array}{ccccc} \dot{x} & \dot{y} & \dot{u} & \dot{v} & \lambda \end{array} \\
\begin{array}{c} f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \end{array} &
\begin{pmatrix}
1 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 0
\end{pmatrix}
\end{array}
$$

No perfect matching

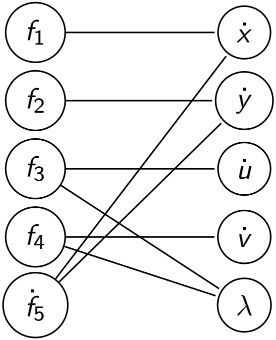
Suppose a consistent initialization

- f_5 is not used
- f_5 holds for all t, then $\dot{f}_5$ has to hold for all t
- $\dot{f}_5 : 2x\dot{x} + 2y\dot{y} = 0$ (symbolic differentiation)

Incidence Matrix

$$\begin{array}{c c} & \begin{array}{ccccc} \dot{x} & \dot{y} & \dot{u} & \dot{v} & \lambda \end{array} \\ \begin{array}{c} f_1 \\ f_2 \\ f_3 \\ f_4 \\ \dot{f}_5 \end{array} & \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

Bipartite Graph



Suppose a consistent initialization

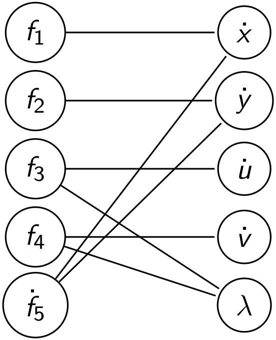
- f_5 is not used
- f_5 holds for all t, then $\dot{f}_5$ has to hold for all t
- $\dot{f}_5 : 2x\dot{x} + 2y\dot{y} = 0$ (symbolic differentiation)

Incidence Matrix

$$
\begin{array}{cc}
 & \begin{array}{ccccc} \dot{x} & \dot{y} & \dot{u} & \dot{v} & \lambda \end{array} \\
\begin{array}{c} f_1 \\ f_2 \\ f_3 \\ f_4 \\ \dot{f}_5 \end{array} &
\begin{pmatrix}
1 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 1 & 1 \\
1 & 1 & 0 & 0 & 0
\end{pmatrix}
\end{array}
$$

Still no matching !

Bipartite Graph

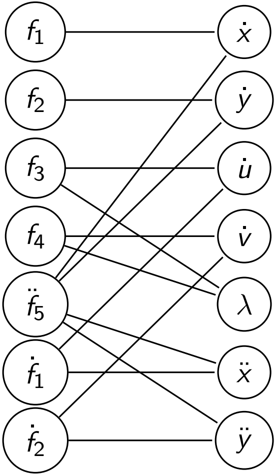


Incidence Matrix

$$
\begin{array}{c} \\ f_1 \\ f_2 \\ f_3 \\ f_4 \\ \ddot{f}_5 \\ \dot{f}_1 \\ \dot{f}_2 \end{array}
\begin{array}{c}
\begin{array}{ccccccc} \dot{x} & \dot{y} & \dot{u} & \dot{v} & \lambda & \ddot{x} & \ddot{y} \end{array} \\
\begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & 1 & 0 & 0 \\
1 & 1 & 0 & 0 & 0 & 1 & 1 \\
0 & 0 & 1 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 1
\end{pmatrix}
\end{array}
$$

Matching Successful!

Bipartite Graph



Implicit Function Theorem

Let $F(X, Y)$: n equations, where $|Y| = n$, $|X| = m$.
If $(u, v) \in \mathbb{R}^{m+n}$ is such that $F(u, v) = 0$ and $J = \frac{\partial F}{\partial Y}$ is nonsingular at (u, v), then there exists, in an open neighborhood U pf u, a unique set of functions G such that $v = G(u)$ and $F(w, G(w)) = 0$ for all $w \in U$

Implicit Function Theorem

Let $F(X, Y)$: n equations, where $|Y| = n$, $|X| = m$.
If $(u, v) \in \mathbb{R}^{m+n}$ is such that $F(u, v) = 0$ and $J = \frac{\partial F}{\partial Y}$ is nonsingular at (u, v), then there exists, in an open neighborhood U pf u, a unique set of functions G such that $v = G(u)$ and $F(w, G(w)) = 0$ for all $w \in U$

Structural NonSingularity

A square matrix is said to be structurally nonsingular if it remains almost everywhere nonsingular when its nonzero coefficients vary over some neighborhood.

Implicit Function Theorem

Let $F(X, Y)$: n equations, where $|Y| = n$, $|X| = m$.
If $(u, v) \in \mathbb{R}^{m+n}$ is such that $F(u, v) = 0$ and $J = \frac{\partial F}{\partial Y}$ is nonsingular at (u, v), then there exists, in an open neighborhood U pf u, a unique set of functions G such that $v = G(u)$ and $F(w, G(w)) = 0$ for all $w \in U$

Structural NonSingularity

A square matrix is said to be structurally nonsingular if it remains almost everywhere nonsingular when its nonzero coefficients vary over some neighborhood.

Relation to the BLT Decomposition

The Jacobian $J = \frac{\partial F}{\partial Y}$ is structurally nonsingular if and only if G_F (the incidence graph related to F) can be decomposed in a BLT form.

Pantelides algorithm (1988) attempts to decompose the function F of a given DAE into a BLT form by exhibiting latent equations.

- First structural analysis of DAE
- Not guaranteed to terminate
- Applies only to first-order systems
- May overestimate the differential index
- Other methods: Signature (Σ) method, J. D. Pryce (2001)

Index Reduction

- Given a DAE $F(x, \dot{x}, t)$, we have seen how to perform a structural analysis to numerically compute $\dot{x}$ function of x. The structural nonsingularity ensures that, generically, one can perform the computation following the block order suggested by the BLT decomposition. Thus, one is able to compute the numerical values of the derivatives given a consistent state of the system and carry on with a standard numerical integration.

- Note: the **structural index** is not always equal to the **differentiation index** (see the first reference below for concrete examples).

- Guangning Tan, Ned S. Nedialkov, John D. Pryce, *Symbolic-Numeric Methods for Improving Structural Analysis of Differential-Algebraic Equation Systems*, Book Chapter in "Mathematical and Computational Approaches in Advancing Modern Science and Engineering", pp. 763-773, Springer, 2016 (available on arXiv:1505.03445 [cs.SC])
- Constantinos C. Pantelides, *The Consistent Initialization of Differential-Algebraic Systems*, SIAM J. Sci. and Stat. Comput. Volume 9, Issue 2, pp. 213-231 (1988)
- Rajeev Motwani, *Average-case Analysis of Algorithms for Matchings and Related Problems*, Journal of the ACM, 41 (6): pp. 1329-1356 (1994)
- Tarjan, R. E., *Depth-first search and linear graph algorithms*, SIAM Journal on Computing, 1 (2): pp. 146-160 (1972)