Algorithmics

Sebastian Iwanowski
FH Wedel

4. Graph algorithms
4.4 Computation of graph matchings

Algorithmics 4

Matchings in graphs

Def.: A matching is a set of edges such that no edge is adjacent to another edge.

Def.: maximum matching:

i) maximum number of edges (only this is investigated in the references below)

ii) for valued edges: matching with maximum value

Def.: Set theoretic statement of graph matching (**2DM**):

Given a set $E \subseteq VxV$: Find a maximal subset $T \subseteq E$ where: All elements of T are pairwise disjoint.

Def.: Generalization of graph matching (**kDM**):

Given an set $E \subseteq Vx...xV$: Find a maximal subset $T \subseteq E$ where: All elements of T are pairwise disjoint.

Theorem: kDM is NP-complete for $k \geq 3$ and 2DM is in P.

References:

Alt, Definition 4.6.1

Laszlo Lovasz / Michael Plummer: *Matching Theory*, North Holland 1986, ISBN 9630541688, ch. 9.1
James McHugh: *Algorithmic Graph Theory*, Prentice Hall 1990, ISBN 0130236152, ch. 8.3
Christos Papadimitriou / Kenneth Steiglitz: *Combinatorial Optimization*, Dover 1998, ch. 10

Algorithmics 4

Matchings in graphs (maximum number of edges)

Special case considered in detail: Matchings in bipartite graphs

Def.: A flow f is integer-valued ⇔ f(u,v) is integer-valued *for each edge* (u,v)

Def.: For a given bipartite Graph G = ((V,U),E), build an s/t-network G‘ with properties:
There is a source s ∈ G‘ with a directed edge *to* each vertex of V, each edge having capacity 1.
There is a target t ∈ G‘ with a directed edge *from* each vertex of U, each edge having capacity 1.
For each edge of G, there is a directed edge from a vertex in V to a vertex in U having capacity 1.

Lemma: (Alt 4.6.3) G has got a matching where |M|=k ⇔ G‘ has got an integer valued flow where |f| = k

Theorem: (Alt 4.6.4, Cormen 26.11) Given an s/t-network with integer capacities for all edges:
i) Then the value of a maximum flow is an integer as well.
ii) There exists always a maximum flow that is integer-valued.

Proof: i) follows from max flow / min cut theorem
ii) has to be proven separately

Corollary: The maximum matching in G is one-to-one related to the maximum matching in G‘.

References:

Alt, Kap. 4.6
Cormen, ch. 26.3 (maximum bipartite matching)

Matchings in graphs (maximum number of edges)

Algorithms for bipartite matchings und integer valued flows

Prop.: A maximum bipartite matching can be found by the maximum flow algorithm of Edmonds-Karp in O(nm).
(Remark: For integer-valued networks, time complexity is better than for arbitrary networks).

Improvements:

Hopcroft-Karp: $O(n^{0,5}m)$
Alt et al.: $O(n^{1,5}(m/\log n)^{0,5})$ (this is an improvement for dense graphs)

Prop.: In unit networks (networks where each edge has got capacity 1), the algorithm of Dinic needs only $n^{0,5}$ iterations.
The inner operations do not sum up to O(nm) as in the general case, but only to O(m).
Thus, the algorithm of Dinic performed in unit networks requires run time $O(n^{0,5}m)$.

Corollary: The run time of Hopcroft-Karp for bipartite matching may be achieved also with the algorithm of Dinic.

References:

Alt, Kap. 4.7
Cormen, Problem 26-7
Turau Kap. 7 (vor allem Literaturhinweise 7.6)

Matchings in graphs (maximum number of edges)

Techniques for matchings in general graphs

Def.: An augmenting path is a path from an unmatched vertex to an unmatched vertex using unmatched and matched edges alternately.

Def.: An outer vertex of an augmenting path is a vertex being an odd successor in the path, i.e., it is the 1., 3., 5., ... vertex of the augmenting path.
Except for the first, an outer vertex is always at the end of a matched edge.

Def.: A blossom is an uneven circle with a maximum matching:
A blossom consists of 2k+1 edges, k being matched.

(good examples in McHugh)

Remark: A blossom will be discovered in the course of the search for augmenting paths whenever the fact is discovered that two outer edges are adjacent.

References for details:

Laszlo Lovasz / Michael Plummer: *Matching Theory*, North Holland 1986, ISBN 9630541688, ch. 9.1
James McHugh: *Algorithmic Graph Theory*, Prentice Hall 1990, ISBN 0130236152, ch. 8.3
Christos Papadimitriou / Kenneth Steiglitz: *Combinatorial Optimization*, Dover 1998, ch. 10

Algorithmics 4

Matchings in graphs (maximum number of edges)

Algorithm of Edmonds for general graphs

Main loop: 1) Search for augmenting path AP:

1a) Start with an unmatched vertex and an empty augmenting path AP.

1b) Look at neighbors:

If one is not matched -> augmenting path AP is found.

Otherwise, augment AP by an edge to a neighbor and its matched vertex:

If this yields a blossom,

contract the blossom and continue with the contracted graph

2) If no augmenting path AP has been found -> Matching is maximum.

If yes:

2a) Decontract graph by all previously found blossoms.

2b) Augment AP in original (decontracted) graph.

2c) Increase matching by inverting the matching of AP and continue at 1)

References for details:

Laszlo Lovasz / Michael Plummer: *Matching Theory*, North Holland 1986, ISBN 9630541688, ch. 9.1

James McHugh: *Algorithmic Graph Theory*, Prentice Hall 1990, ISBN 0130236152, ch. 8.3

Christos Papadimitriou / Kenneth Steiglitz: *Combinatorial Optimization*, Dover 1998, ch. 10

Matchings in graphs (maximum number of edges)

Algorithm of Edmonds for general graphs

Time complexity: 1) $O(n^2)$ (proof nontrivial)

2) $O(n)$ (clear)

The main loop is performed $O(n)$ times, because each time the matching is increased by one edge -> total time complexity: $O(n^3)$

Correctness:

Prop. 1: Matching is maximum ⇔ There is no augmenting path

Proof: Homework

Prop. 2: Let M be a matching in G. Let G have a blossom and let G‘ be the contracted graph:
G has got an augmenting path for M ⇔ G‘ has got an augmenting path for M

References for details:

Laszlo Lovasz / Michael Plummer: *Matching Theory*, North Holland 1986, ISBN 9630541688, ch. 9.1
James McHugh: *Algorithmic Graph Theory*, Prentice Hall 1990, ISBN 0130236152, ch. 8.3
Christos Papadimitriou / Kenneth Steiglitz: *Combinatorial Optimization*, Dover 1998, ch. 10