Combinatorics

Exercise 4 – Edmonds' algorithm

Homework

Voluntary homework for bonus points
Deadline: **10. 11. 2021 9:00**
Send to: matej@kam.mff.cuni.cz (in PDF, preferably LaTeX, of course)

1. Analyze Edmonds' algorithm from the course lecture notes and prove that it runs in polynomial time.[1]

   (a) Find an algorithm which finds Edmonds forest in polynomial time.

   (b) Observe that Step 2 of *IsMaximumMatching* can be implemented in polynomial time.

   (c) Show how Step 3 of *IsMaximumMatching* can be implemented in polynomial time (except for the recursive call to $IsMaximumMatching(\widetilde{M}, \widetilde{G})$).

   (d) Let $T(n)$ be an upper bound on time in which *IsMaximumMatching* runs on any graph on n vertices. Use the above points to show that $T(n) \leq T(n-2)+p(n)$, where $p(n)$ is some polynomial in n. Use that to conclude that $T(n)$ itself is a polynomial in n.

   (e) Conclude that FindMaximumMatching itself called on a graph on n vertices itself does as most $q(n)$ steps when called on a graph on n vertices, where $q(n)$ is polynomial in n.

Problems

1. For every $k \geq 1$, build a graph with minimum degree at least k and exactly one perfect matching.
2. Prove that if G has no perfect matching, then G has a vertex v such that all edges incident with v belong to some maximum matching (each to a different one of course).
3. Construct a 3-regular graph with no perfect matching.
4. Decide if the following statement is true or false: If G is a 3-regular graph with no bridges, then there are six perfect matchings $M_1, \dots, M_6$ in G such that every edge of G is contained in exactly two of them.

[1] If implemented for example in Pascal, C, C++ etc., we will have discussed this at the tutorial session, but feel free to ask questions.