MOPSI

Léo Butsanets & Mohammed Hamzaoui
Supervised by: Julien Guyon & Fredr

January 2023

1 Existence - General Version

We are given a graph $G = (V, E)$, where the vertices are partitioned into k subsets $V_1, \ldots, V_k$. An integer d is given. The goal is to decide whether there exists a subset F of E such that every vertex has d neighbors in F within each V_i.

This problem is polynomial, even if d is free (i.e., complexity of the form $O(|V|^a \log^b(d))$). Indeed, the problem is "separable":

- First problem: for each pair of distinct integers i, j from $[k]$, decide whether there exists a subset of edges in the bipartite graph $(V_i \cup V_j, E_{ij})$ giving each vertex a degree of d, where E_{ij} is the subset of E with one endpoint in V_i and the other in V_j. This can be done in polynomial time using a "flow algorithm".
- Second problem: for each integer i from $[k]$, decide whether there exists a subset of edges in the "induced" subgraph $G[V_i] = (V_i, E_i)$ giving each vertex a degree of d, where E_i is the subset of E with both endpoints in V_i. This can be done in polynomial time using an algorithm for "simple b-matchings". This is a challenging algorithm.
- We have a solution if and only if the answer is "yes" for the $\frac{n(n-1)}{2} + n = \frac{n(n+1)}{2}$ previous tests.

1.1 First Problem

Input: $G = (U \cup V, E)$ bipartite graph, d
Output: Find a subset $F \subseteq E$ such that $\forall v \in V$, $deg_F(v) = d$.

This can be seen as a maximum flow problem. Consider $G = (U \cup V \cup \{s,t\}, E \cup \{(s,u), (v,t) | u \in U, v \in V\})$ with capacities:

- $\forall (u, v) \in U \times V$, $c(u, v) = 1$
- $\forall u \in U$, $c(s, u) = d$

- $\forall v \in V$, $c(v, t) = d$

Algorithm:

- Linear programming: The constraints are given by admissible flow, where admissible flow is a flow that does not exceed the capacity of an arc. Maximize $\sum_{v \in V} f(s, v)$.
- Ford-Fulkerson Algorithm: While there exists a path between the source and sink in the residual graph, send the minimum of residual capacities along this path. Complexity is $O(E \cdot maxflow)$ in the integer case.
- Edmonds-Karp Algorithm: Specialization of the Ford-Fulkerson algorithm, where augmenting paths are the shortest paths (in the number of arcs) in the residual graph (using breadth-first search). Complexity is $O(VE^2)$.

1.2 Second Problem

Input: $G = (V, E)$, $d \in \mathbf{N}$
Output: Find a simple-d-matching.

1.3 Case $d = 1$

In the case of $d = 1$, the problem is to find a perfect matching in a graph. Edmonds' algorithm for matchings (blossom algorithm), also known as the algorithm of blossoms and petals, is an algorithm for constructing maximal matchings on graphs. The algorithm has a time complexity of $O(|E||V|^2)$.

1.4 Case d Free

1.5 Construction of a New Graph

Consider the graph constructed from $G = (V, E)$ as follows:

- Replace each vertex u by d copies $u_1, \ldots, u_d$.
- For each edge (u, v) (which is removed), create two vertices u' and v' connected by an edge, add d edges between the u_i's and u', add d edges between the v_i's and v'. Let the new graph be $G' = (d|V| + 2|E|, (2d + 1)|E|)$.

1.6 Connection between $d = 1$ and Free d

Lemma: There exists a simple-d-matching in G if and only if there exists a perfect matching in the graph G' explicitly detailed above.

(To be proven)