remote
9th LAWCG
and
MDA
November 25th, 2020

Connected Matchings

Masquio, Bruno P. – IME/UERJ – brunomasquio@ime.uerj.br, Pinto, Paulo E. D. – IME/UERJ – pauloedp@ime.uerj.br, Szwarcfiter, Jayme L. – COPPE/UFRJ – jayme@nce.ufrj.br

Introduction

Graph matching problems are well known and studied, in which we want to find sets of pairwise non-adjacent edges[1]. This work focus on the study of matchings that induce subgraphs with special properties [2][3]. For this work, we consider the property of being connected, also studying it for weighted or unweighted graphs. For unweighted graphs, we want to obtain a matching with the maximum cardinality, while, for the weighted graphs, we look for a matching whose sum of the edge weights is maximum.

Objective

The problem of maximum connected matching is polynomial[1]. We show ideas that lead to two linear algorithms. One of them, having a maximum matching as input, determines a maximum unweighted connected matching. The complexity of the maximum weighted connected matching problem is unknown for general graphs. However, we present a linear time algorithm that solves it for trees.

Unweighted Connected Matchings

For a graph G and a matching M, we denote $G[M]$ as the subgraph induced by the vertices of M and $N(v)$ as the set of neighbors of v in G. Note that, in the same graph, the cardinalities of a maximum unweighted connected matching and of a maximum weighted connected matching are not always the same. We exemplify in Figure 1. Therefore, we expect that these problems have different computational treatments.

Theorem 1

If G is connected and M is an unweighted maximum matching in G, then the unweighted maximum connected matching has cardinality $|M|$ [2]

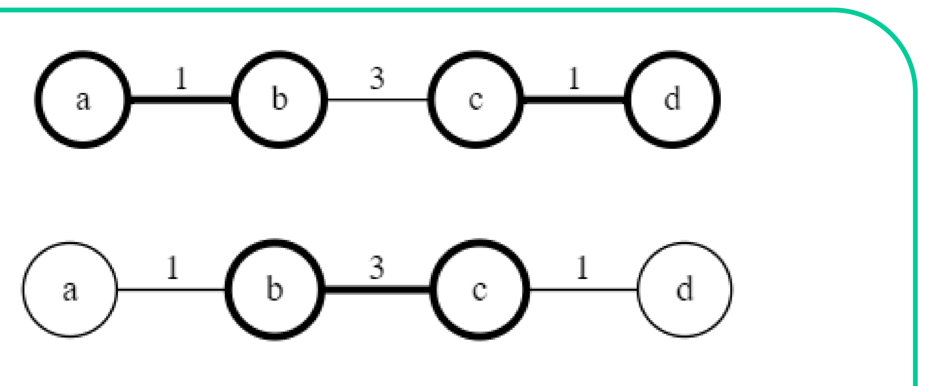

Figure 1: Two maximum connected matchings of a graph.

The proof of Theorem 1[2] is based on the fact that, in a graph G, if M is a maximum matching and $G[M]$ is disconnected, in which C is connected component of $G[M]$, then it is possible to redefine the edges of M in order to increment vertices of C in M, successively, until $G[M]$ has a single component.

We present an idea to do all this process and leave $G[M]$ connected in linear time. Let M be a maximum matching such that $G[M]$ is disconnected and r a M-saturated vertex. Consider C_r to be the component of $G[M]$ which contains r. We use two sets, Q_s and Q_n, to store M-saturated and M-unsaturated vertices, respectively. Additionally, we employ a set C, to which vertices of C_r or new vertices are added. A main loop can be executed until $G[M]$ equals C. Each iteration is divided into two other auxiliary loops and includes at least one vertex at C. The first auxiliary loop, for each vertex v of Q_s, analyzes $N(v)$, and properly adds to this set each vertex of that neighborhood that has not yet entered the set. The second auxiliary loop, for each vertex v of Q_n, if $w \in N(v) \setminus C$ exists, then w is saturated by some edge, (w,u), and we perform the *edge exchange* operation in M. Such operation removes (w,u) and adds the edge (v,w) to M. In the end of this process, $G[M]$ will be connected.

Weighted Connected Matchings

Though it is still unknown the complexity of finding maximum weighted connected matchings, we present an idea that leads to a linear solution for trees. Let T be a tree and r, $v \in V(T)$. We denote T^r as a tree T rooted in r and T^r_v as the subtree of T^r rooted in v. Also, $S(r,v)$ is the set of all sons of v in T^r_v and $weight(v,w)$ is the weight of the edge (v,w).

Theorem 2

Let G be a connected graph and M a maximum connected macthing of G. Then M saturates all articulations in G

Without loss of generality, by Theorem 2, we know that, for a tree T, each articulation v must be saturated. We look for the neighbors of v, which maximize the weighted sum of the edges to build a maximum connected matching in T^r_v. For such a construction, we consider r as any vertex of T, and apply a dynamic programming algorithm described below. We define the sum of the edge weights of a maximum weighted matching in T^r_v as B_v if v is matched with one of its children, and $\overline{B_v}$ if v is matched with its father. We can determine this variables as follows. If v is a leaf, then $B_v = \overline{B_v} = 0$. Else, consider the following equations.

$$\overline{B_v} = \sum_{u\in S(r,v)} B_u$$

$$B_v = \max_{a\in S(r,v)} \left(\overline{B_a} + weight(a,v) + \sum_{u\in S(r,v)\setminus\{a\}} B_u \right)$$

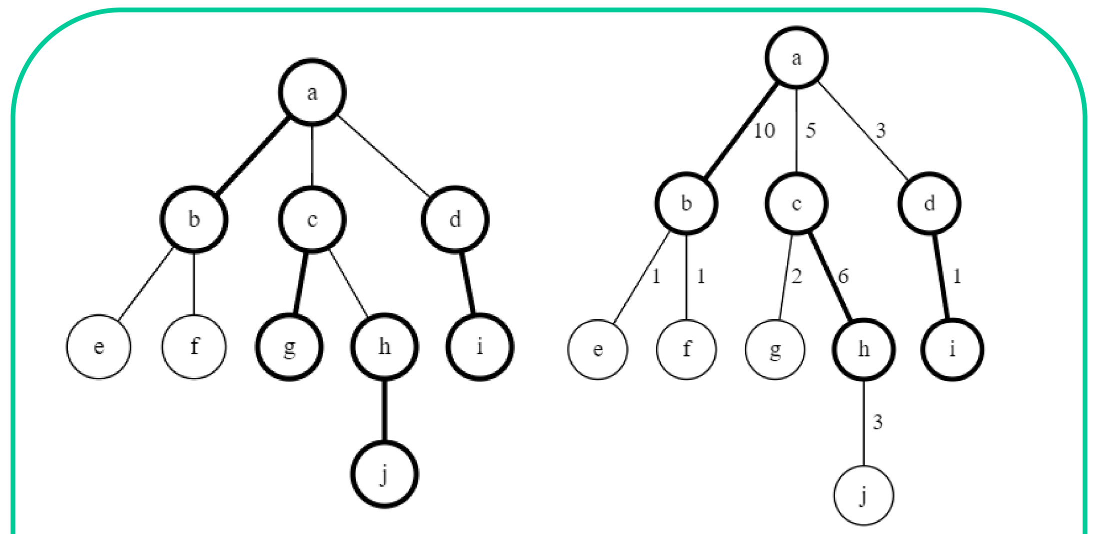

Figure 2: Two maximum connected matchings. The left matching refers to a unweighted while the right, to weighted.

An algorithm can dynamically build a maximum connected matching M as follows. From an arbitrary articulation r elected as root, two searches are made. The first computes the vertices from the leaves to the root r. It obtains, for each vertex u, a child vertex s_u of u that maximizes B_u. In addition, $\overline{B_u}$ is calculated from the sum of B_w for all its children w. The second search is responsible for building M, computing the vertices from r to the leaves, so that, when a vertex u is processed, if u is not part of M yet, we add (s_u, u) to M. In the end, M will be a maximum weighted connected matching.

References

[1] MICALI, S. and VAZIRANI, V. V.. An $O(\sqrt{|V|}|E|)$ algorithm for finding maximum matching in general graphs. In 21st Ann. Symp. on Foundations of Comp. Sc., pages 17–27, 1980.

[2] GODDARD, W., HEDETNIEMI, S. M., HEDETNIEMI, S. T., and LASKAR, R.. Generalized subgraph-restricted matchings in graphs. Discrete Math., 293 (2005), 129-138.

[3] MASQUIO, B. P.. Emparelhamentos Desconexos. Master's Thesis, Universidade do Estado do Rio de Janeiro. Available in https://github.com/BMasquio/papers/raw/master/MastersThesisBrunoMasquio.pdf, 2019.

Acknowledgment

UERJ
CAPES
FAPERJ
CNPq