Entrega Puntuable

Víctor Alcázar
Albert Ribes
Kosmas Palios

June 1, 2017

Contents

1 Statement

We've been given the following problem to talk about.

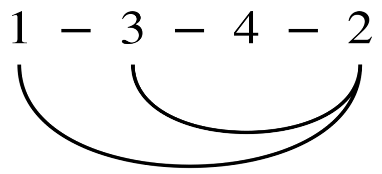
Minimum cut linear arrangement

Given a graph $G = (V, E)$, compute a one-to-one function $f : V \to [1..|V|]$ so that the maximum number of cut edges in any integer point is minimised, i.e.

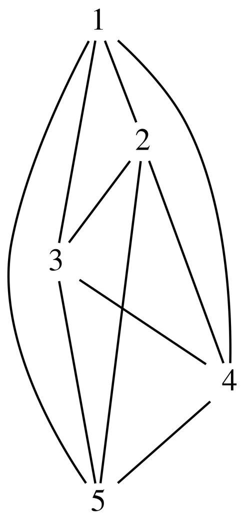
$$\max_{i \in 1..|V|} |\{\{u, v\} \in E : f(u) \leq i < f(v)\}|$$

2 Example Instances

In order to visualize the problem and attack it using more intuition, we must consider that the cuts at integer points can be seen easily in the following manner. Given a graph G and an ordering function f, we draw the nodes in an horizontal line, in the order given by f. What we are trying to minimize is the size maximum cut among the S_i's , where $S_i = \{v \in V : f(v) \leq i\}$, for $i = 1, 2, \ldots, n-1$. But on the graph drawing mentioned above, the size of every S_i is the number of edges existing in the space between the vertical lines passing by nodes i and i+1. To give an example, in the following linear arrangement we easily deduce that the minimum S_i size is 3:

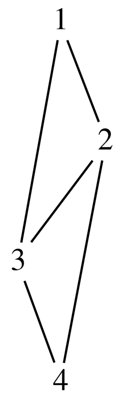


The problem instances are Graphs G. As an example, we take $G = K_5$:

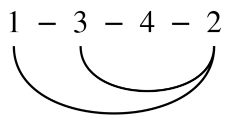


Here, every node is connected to every one of the rest. Therefore, every node is identical to every one else. In this case, every assignment of numbers $1 \ldots n$ to the nodes yields the same graph. Therefore to solve this instance, we consider a random labeling of our nodes, and we compute the cuts at all integer points between 1 and n.

Another instance, could be this graph.



Here, the assignment matters. In order to attack the problem in a more intuitive manner, it would be more suitable to have visualize the problem as follows: Given graph G, create a drawing of G such that the nodes are placed on a straight horizontal line. The objective is to minimize the maximum number of edges that exist in the space between two consecutive nodes. For example, the graph below is a visualization of the assignment $f(1) = 1, f(2) = 4, f(3) = 2, f(4) = 3$, which gives us $max(S_i) = 3$.



By the way, the above assignment gives us the best possible results, w.r.t. the requirements of the problems. That is, there is no assingment that gives us a lower maximum cutwidth.

3 Decisional Problem

Given a graph $G = (V, E)$ and $k \in N$, say if there exists one ordering of the vertexs so that the maximum number of cut edges in any integer point is less than or equal k, i.e.

$$\exists f : V \to [1..|V|]| \max_{i \in 1..|V|} |\{\{u, v\} \in E : f(u) \leq i < f(v)\}| \leq k$$

4 Parametrized Problems

Using this problem, and combining it with a computable function $\kappa : \Sigma^* \to N$ we can formulate a Parametrized version of this problem.

An example:

Input	Graph G and tw = treewidth(G)
Paremeters	tw
Question	What is the Min-Cut Linear Arrangement of G?

Given that we find an expression Monadic Second Order Logic, the above can be solved in $O(n * f(tw))$ [1].

Obviously, one parametrization that makes our problem FPT, is the following:

Input	Graph G
Paremeters	V(G)
Question	What is the Min-Cut Linear Arrangement of G?

Here we can use our bruteforce algorithm on the possible linear arrangements of G, which runs in $V(G)! * (V(G) * E(G)$ time, where $V(G)!$ is to create every possible linear arrangement and $(V(G) * E(G)$ is to compute a cutwidth of a given arrangement. So in this case we have $f(\kappa) * E(G)$ time.

5 A ($log^2(n)$) Approximation

There exists an algorithm that runs in subexponential time and achieves an ($log^2(n)$) approximation of the min.cut linear arrangement problem [2].

Basic elements used are:

- An algorithm that computes in subexponential time a ($\frac{1}{3}$)-balanced cut whose capacity is within an $O(nlogn)$ factor of the capacity of a minimum bisection cut. It is described in [2].
- The idea that the capacity of a minimum bisection cut is lower bound on the optimal arrangement.

The second element can be understood better considering that the cut in the middle of the linear arrangement is a bisection cut, and in every linear arrangement there is a bisection cut. So, in whatever order may we put the vertices in the line, we can't have cutwidth smaller than the capacity of a minimum bisection cut.

Combining the 2 elements above, we can get like the following algorithm:

Find a (1/3)-balanced cut in G V , say (S, S'), and recursively find a numbering of S in G_S using numbers from 1 to —S— and a numbering of S' in $G_{S'}$ using numbers from —S— + 1 to n. Of course, the recursion ends when the set is a singleton, in which case the prescribed number is assigned to this vertex

It is claimed that this algorithm runs in logn * (time of the 13-balanced cut algorithm) and achieves an $O(log^2(n)$ factor for the minimum cut linear arrangement problem.

6 Various results

In this section we will present a number of results regarding our problem.

The problem for genera1 graphs is NP-complete [4], [5], even for planar graphs [8] For fixed cost k, the problem can be solved in $O(n^{k-1})$ time [9]. For trees, better results have been found.

The restriction of the problem to trees has been open for some time. In 1982 Lengauer described an approximation algorithm for trees in [6] that produces a layout with cutwidth at most twice the optimal. He also determined exactly the cutwidth of complete k-ary trees. M. Chung et al. present in [7] an algorithm that solves the MINCUT problem on trees in time $O(n(logn)^{d-2})$ where d is the maximum degree of the tree. Thus, their algorithm works in polynomial time for bounded degree trees, but exponential time in general. Yanakakis in 1983, presents an algorithm that finds a min-cut linear arrangement of a tree in O(n log n) time [3].

References

[1] Maria Serna, *Ampliació d'Algorismia, Class transparencies*, Fall 2017

[2] Vijay Vazirani *Approximation Algorithms*, 2001, Springer

[3] Mihalis Yannakakis *A Polynomial Algorithm for the Min-Cut Linear Arrangement of Trees* 1985, AT&T Bell Laboratories, Murray Hill, New Jersey

[4] Garey, M. R., AND Johnson, D. S. *Computers and Intractability: A Guide to the Theory of NP-completeness.* W. H. Freeman, San Francisco, Calif., 1979.

[5] Gavril, F *Some NP-complete problems on graphs*, In Proceedings of the 11th Conference on Information Sciences and Systems. Johns Hopkins University, Baltimore, Md., 1977, pp. 91-95

[6] Lenaguer, T. U *Upper and lower bounds on the complexity of the min-cut linear arrangement on trees* Siam J. Alg. Disc. Meth. 3 (1982), 99-113

[7] Chung, M., MAKEDON, F., Sudborough, .I.H. and Turner, J. *Polynomial tile algorithms for the min cut problem on degree restricted trees* Proceedings of the 23rd Annual Symposium on the Foundations of Computer Science. IEEE, New York, 1982, pp. 262-271

[8] Monien, B. and Sudborough, I. H. *Min Cut is NP-complete for Edge Weighted Trees* Unpublished manuscript (1985)

[9] Makedon, F. *Layout Problems and Their Complexity* Ph.D. Thesis, EECS Department, Northwestern University (1982)