Information Processing Letters 115 (2015) 913–916

Contents lists available at ScienceDirect

Information Processing Letters

www.elsevier.com/locate/ipl

A faster algorithm for the cluster editing problem on proper interval graphs

Min Chih Lin [a,*], Francisco J. Soulignac [b], Jayme L. Szwarcfiter [c,d]

[a] CONICET and Departamento de Computación, Universidad de Buenos Aires, Argentina
[b] CONICET and Departamento de Ciencia y Tecnología, Universidad Nacional de Quilmes, Argentina
[c] Instituto de Matemática, NCE and COPPE, Universidade Federal do Rio de Janeiro, Brazil
[d] Instituto Nacional de Metrologia, Qualidade e Tecnologia, Brazil

ARTICLE INFO

Article history:
Received 4 December 2014
Received in revised form 21 July 2015
Accepted 21 July 2015
Available online 26 July 2015
Communicated by B. Doerr

Keywords:
Graph algorithms
Cluster editing problem
Proper interval models
Linear space algorithm

ABSTRACT

We develop a linear-space $O(n+m)$ time algorithm to solve the cluster editing problem for proper interval models, where n and m are the number of vertices and edges of the represented graph.

1. Introduction

Data clustering based on graphs is a relevant topic with applications in different areas, for instance, computational biology. In general terms, the problem consists of transforming a graph into a set of *clusters*, that is a subset of vertices similar in some way. The process of transformation consists of removing or adding edges to the original graph. In general, the term *edge edit* is employed to mean either an edge removal or addition.

The most common and extensively studied clustering problem is that of transforming a graph into a set of disjoint cliques, by edge edits. In this case, each cluster corresponds to a disjoint clique. A natural parameter to measure the effort of this transformation is that of counting the number of edits. The corresponding Cluster editing problem, whose goal is to determine whether an input graph G can be transformed into a set of disjoint cliques by at most k edge edits, was proven NP-hard by Křivánek and Morávek [1] (see also [2]).

In spite of its hardness, the problem was proved to be fixed-parameter tractable, a result that follows from [3]. It admits a kernel having at most $2k$ vertices, which can be constructed in $O(nm)$ time, employing an algorithm by Chen and Meng [4]. Alternatively, a kernel with $O(k^2)$ vertices can be obtained in $O(n+m)$ time [5]. Other possible parameterizations have been considered by Damaschke [6] and Komusiewicz and Uhlmann [7]. An exact algorithm of complexity $O^*(1.62^k)$ has been described by Böcker [8]. ILP formulations of the Cluster editing problem have been proposed by Böcker et al. [9], while heuristics have recently been described by Bastos et al. [10]. A generalization of the original cluster editing so as to allow overlaps among the clusters has been considered by Damaschke [11] and Fellows et al. [12]. There is also an extensive literature on the related clustering problem, where only edge removals are allowed (e.g. [7]). See [13] for a recent review, with several complexity results.

* Corresponding author.
E-mail addresses: oscarlin@dc.uba.ar (M.C. Lin), francisco.soulignac@unq.edu.ar (F.J. Soulignac), jayme@nce.ufrj.br (J.L. Szwarcfiter).

http://dx.doi.org/10.1016/j.ipl.2015.07.009
0020-0190/

The extensive literature on FPT algorithms for solving the Cluster editing problem apparently does not extend in the same degree to polynomial time solvable cases. One of these polynomial time cases is that of proper interval graphs. Mannaa [14] described an algorithm of time and space complexity $O(n^2)$ for the latter class.

The reason why proper interval graphs are of interest for the Cluster editing problem is because each vertex of a proper interval graph can be associated to a point of the real line in such a way that two vertices are adjacent if and only if their corresponding points are at distance at most one (see [15]). So, similarity between vertices is expressed by the distance of their corresponding points.

In the present paper, we describe a variation of Mannaa's dynamic programming algorithm, which reduces its complexity to $O(n+m)$ time and $O(n)$ space.

A *cluster graph* C is a graph whose components $C_1, \ldots, C_k$ are cliques. Every cluster graph C is uniquely *represented* by the family $\mathbb{C} = \{C_1, \ldots, C_k\}$. A *clustering* of a graph G is simply a partition of its vertex set; each member of a clustering $\mathbb{C}$ is a *cluster* of $\mathbb{C}$. The *value* of $\mathbb{C}$ is the number $\mathrm{val}(\mathbb{C})$ of edits required to transform G into the cluster graph represented by $\mathbb{C}$. The *optimal value* for G is $\mathrm{opt}(G) = \min\{\mathrm{val}(\mathbb{C}) \mid \mathbb{C} \text{ is a clustering of } G\}$. Those clusterings of G with value $\mathrm{opt}(G)$ are referred to as *optimal*. With this terminology, Cluster editing is the problem of finding an optimal clustering of G.

We use a convenient representation of proper interval graphs (see Fig. 1). A *rightmost function (of order n)* is a nondecreasing function $r\colon \{1, \ldots, n\} \to \{1, \ldots, n\}$ such that $r(i) \geq i$ for every $1 \leq i \leq n$; note that $r(n) = n$. The *leftmost function* associated to r is the nondecreasing function $\ell\colon \{1, \ldots, n\} \to \{1, \ldots, n\}$ such that $\ell(i) = \min\{j \mid i \leq r(j)\}$ for every $1 \leq i \leq n$; note that $\ell(1) = 1$ and $\ell(i) \leq i$ for every $1 < i \leq n$. Each rightmost function r defines a graph $G(r)$ with vertex set $\{v_1, \ldots, v_n\}$ where v_i is adjacent to $v_j \neq v_i$ if and only if $\ell(i) \leq j \leq r(i)$. A graph G is a *proper interval (PIG)* graph when it is isomorphic to $G(r)$ for some rightmost function r. In such a case, we say that G *admits* (ℓ, r), whereas (ℓ, r) is a *straight representation* of G. For the sake of notation, we assume the vertices of a PIG graph are always $v_1, \ldots, v_n$. It is not hard to see that a graph is PIG if and only if it is the intersection graph of a family of inclusion-free intervals on the real line (see Fig. 1 or [16]). In fact, the recognition algorithm by Deng et al. [16] outputs a straight representation for any input PIG graph.

Let G be a graph with a straight representation (ℓ, r). We write $d(j) = j - \ell(j)$ for every $1 \leq j \leq n$, i.e., $d(j)$ is the number of neighbors of v_j in $v_1, \ldots, v_{j-1}$. For $0 \leq i \leq j \leq n$, we denote by $G_{i,j}$ the subgraph of G induced by $v_{i+1}, \ldots, v_j$, while $m_{i,j}$ is the number of edges of $G_{i,j}$. (We exclude v_i from $G_{i,j}$ to simplify index manipulation below.) For the special case $i = 0$, we write $G_j = G_{0,j}$ and $m_j = m_{0,j}$, so $m = m_n$. Note that G_0 is an *empty* graph that has no vertices and $m_0 = 0$ edges; we need G_0 in order to deal with the base case of the recurrence relation defining the dynamic programming algorithm.

2. The algorithm

In this section we improve Mannaa's algorithm by reducing its time and space complexity. Both the algorithm by Mannaa and our implementation are based on the following theorem, which has been originally conjectured by Damaschke, see [14].

Theorem 1. *(See [14].) Every PIG graph admits an optimal clustering in which:*

each cluster consists of $v_{i+1}, \ldots, v_j$

for some $0 \leq i < j \leq n$. (cons)

Remark 1. A word of caution is required here. As defined in Section 1, the ordering $v_1, \ldots, v_n$ depends on which straight representation (ℓ, r) of G is taken. The article by Mannaa, however, is described in terms of some restricted PIG models. Every PIG model defines a so-called *PIG ordering* of the vertices of G. The original version of Theorem 1 [14] holds only for those PIG orders defined by *unitary* PIG models. However, Roberts [15] proved that every connected PIG graph admits at most two PIG orderings, one the reverse of the other, and it is a well known fact that $v_1, \ldots, v_n$ is a PIG ordering of G (e.g. [16]). Thus, Theorem 1 holds for $v_1, \ldots, v_n$ as well. □

For $0 \leq i \leq j \leq n$, let:

- $\overline{m}_{i,j} = \binom{j-i}{2} - m_{i,j}$ be the number of non-edges of $G_{i,j}$, i.e., the number of edges that must be inserted to transform $G_{i,j}$ into a clique, and
- $\mathrm{cut}_{i,j}$ be the number of edges of G joining a vertex in $\{v_1, \ldots, v_i\}$ with a vertex in $\{v_{i+1}, \ldots, v_j\}$, i.e., the number of edges that must be removed so as to disconnect $G_{i,j}$ from G_i.

We can compute $\mathrm{opt}(G) = \mathrm{opt}(G_n)$ with the recurrence relation of the next theorem.

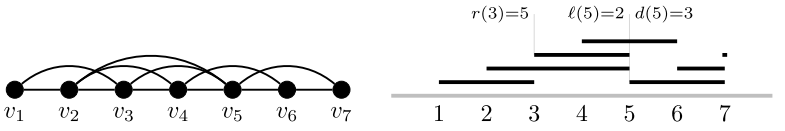
Theorem 2. *For every* $0 \leq j \leq n$

$$\mathrm{opt}(G_j) = \begin{cases} 0 & \text{if } j \leq 1 \\ \min\limits_{j-2d(j)-1 \leq i < j} \{\mathrm{opt}(G_i) + \overline{m}_{i,j} + \mathrm{cut}_{i,j}\} & \text{otherwise} \end{cases} \quad (1)$$

Proof. The theorem is true for $j = 1$; suppose $j > 1$ and let:

- $\mathbb{C}_j$ be an optimal clustering for G_j satisfying (cons), and
- $C = \{v_{i+1}, \ldots, v_j\}$ be the cluster of $\mathbb{C}_j$ that contains v_j.

Consider the clustering $\mathbb{C}$ obtained from $\mathbb{C}_j$ by splitting C into $C \setminus \{v_j\}$ and $\{v_j\}$. (If $j = i + 1$, then $\mathbb{C} = \mathbb{C}_j$.) By definition, v_j is adjacent to $\min\{d(j), j - i - 1\}$ vertices of



(a) $G(r)$ (b) $\{[i, r(i)] \mid 1 \le i \le 7\}$

Fig. 1. (a) The graph $G(r)$ corresponding to the rightmost function $r = \{1 \mapsto 3, 2 \mapsto 5, 3 \mapsto 5, 4 \mapsto 6, 5 \mapsto 7, 6 \mapsto 7, 7 \mapsto 7\}$. The leftmost function ℓ associated to r is $\{1 \mapsto 1, 2 \mapsto 1, 3 \mapsto 1, 4 \mapsto 2, 5 \mapsto 2, 6 \mapsto 4, 7 \mapsto 5\}$. For every rightmost function r, $G(r)$ is the intersection graph of the family of inclusion-free intervals $\{[i, r(i) + i\epsilon] \mid 1 \le i \le n\}$ for a small enough ϵ. (b) Conversely, any family $\mathbb{I}$ of inclusion-free intervals $[s_1, t_1], \ldots, [s_n, t_n]$ with $s_1 < \ldots < s_n$ defines a rightmost function r such that $r(i) = j$ when $s_j < t_i < s_{j+1}$, for every $1 \le i \le n$. If G is the intersection graph of $\mathbb{I}$, then G is isomorphic to $G(r)$.

$C \setminus \{v_j\}$, and it is not adjacent to $\max\{j - i - 1 - d(j), 0\}$ vertices of $C \setminus \{v_j\}$. Thus,

$$\begin{aligned}\mathrm{val}(\mathbb{C}) &= \mathrm{val}(\mathbb{C}_j) + \min\{d(j), j - i - 1\} \\ &\quad - \max\{j - i - 1 - d(j), 0\}.\end{aligned}$$

Since $\mathbb{C}_j$ is optimal, we have $\mathrm{val}(\mathbb{C}) \ge \mathrm{val}(\mathbb{C}_j)$, thus

$$i \ge j - 2d(j) - 1 \tag{2}$$

Next, observe that $\mathbb{C}_i = \mathbb{C}_j \setminus \{C\}$ is a clustering of G_i. Then, by (2), it follows that

$$\begin{aligned}\mathrm{opt}(G_j) &= \mathrm{val}(\mathbb{C}_i) + \overline{m}_{i,j} + \mathrm{cut}_{i,j} \\ &\ge \min_{j-2d(j)-1 \le i < j} \{\mathrm{opt}(G_i) + \overline{m}_{i,j} + \mathrm{cut}_{i,j}\}.\end{aligned}$$

For the other inequality, consider any optimal clustering $\mathbb{C}_k$ of G_k with $j - 2d(j) - 1 \le k < j$. Clearly, $\mathbb{C}_k \cup \{\{v_{k+1}, \ldots, v_j\}\}$ is a clustering of G_j, thus

$$\mathrm{opt}(G_j) \le \min_{j-2d(j)-1 \le i < j} \{\mathrm{opt}(G_i) + \overline{m}_{i,j} + \mathrm{cut}_{i,j}\} \qquad \square$$

The next lemma proves that $\mathrm{opt}(G_j)$ can be obtained efficiently once $\mathrm{opt}(G_i)$ was computed for every $0 \le i < j$.

Lemma 3. *Let $i, j \in \mathbb{N}$ be such that $0 \le i \le j \le n$. If $m_i, \ldots, m_j$ are given, then $\overline{m}_{i,j}$ and $\mathrm{cut}_{i,j}$ can be computed in $O(1)$ time.*

Proof. The lemma is true for $i = 0$ because $m_{0,j} = m_j$ and $\mathrm{cut}_{0,j} = 0$; suppose $i > 0$. By definition,

$$m_{i,j} + \mathrm{cut}_{i,j} = m_j - m_i. \tag{3}$$

To compute $m_{i,j}$ and $\mathrm{cut}_{i,j}$, we observe the following two cases.

Case 1: $j \le r(i)$.
In this case $G_{i,j}$ has all the possible edges, thus

$$m_{i,j} = \binom{j-i}{2}. \tag{4}$$

Then, by (3) we obtain

$$\mathrm{cut}_{i,j} = m_j - m_i - \binom{j-i}{2}. \tag{5}$$

Case 2: $j > r(i)$.
This time, no vertex in $\{v_{r(i)+1}, \ldots, v_j\}$ is adjacent to a vertex in $\{v_1, \ldots, v_i\}$. Then, by (5), it follows that

$$\mathrm{cut}_{i,j} = \mathrm{cut}_{i,r(i)} = m_{r(i)} - m_i - \binom{r(i)-i}{2}. \tag{6}$$

Therefore, by (3), it follows that

$$m_{i,j} = m_j - m_{r(i)} + \binom{r(i)-i}{2}. \tag{7}$$

By (4)–(7), we conclude that $O(1)$ time is enough to compute $m_{i,j}$, $\overline{m}_{i,j}$, and $\mathrm{cut}_{i,j}$ for any $0 \le i \le j \le n$. $\square$

Our algorithm finds $\mathrm{opt}(G_j)$ and m_j for every $j = 0, \ldots, n$; the case $j = 0$ is trivial. For $j > 0$, suppose $\mathrm{opt}(G_{j-2d(j)-1}), \ldots, \mathrm{opt}(G_{j-1})$ and $m_{j-2d(j)-1}, \ldots, m_{j-1}$ have been computed and can be accessed in $O(1)$ time each. Observe, on the one hand, that $m_j = m_{j-1} + d(j) = m_{j-1} + j - \ell(j)$, thus it can be computed in $O(1)$ time. On the other hand, by Theorem 2, $\mathrm{opt}(G_j) = \min_{j-2d(j)-1 \le i < j}\{\mathrm{opt}(G_i) + \overline{m}_{i,j} + \mathrm{cut}_{i,j}\}$, thus $\mathrm{opt}(G_j)$ can be obtained in $O(d(j))$ time by Lemma 3. Consequently, $O(n + m)$ time and $O(\max_{1 \le j \le n} d(j))$ space is required to compute $\mathrm{opt}(G_n)$ when a straight representation (ℓ, r) is given as input.

3. Main theorem

We recall that a straight representation of a PIG graph G can be obtained in $O(n + m)$ time and $O(n)$ space [16]. The main theorem of this note then follows.

Theorem 4. *There exists an $O(n + m)$ time and $O(n)$ space algorithm that solves* Cluster editing *for PIG graphs.*

When all the edges of G are given as part of the input, our algorithm has linear time complexity. It remains an open problem if we can solve Cluster editing in $O(n)$ time for a PIG graph when the straight representation is known.

Acknowledgements

We thank the anonymous reviewers for their helpful comments.

The first author was partially supported by UBACyT Grant 20020120100058, and PICT ANPCyT Grants 2010-1970 and 2013-2205. The second author was partially supported by PICT ANPCyT Grants 2010-1970 and

2013-2205. The third author was partially supported by CNPq and CAPES, research agencies.

References

[1] M. Křivánek, J. Morávek, NP-hard problems in hierarchical-tree clustering, Acta Inform. 23 (3) (1986) 311–323, http://dx.doi.org/10.1007/BF00289116.

[2] R. Shamir, R. Sharan, D. Tsur, Cluster graph modification problems, Discrete Appl. Math. 144 (1–2) (2004) 173–182, http://dx.doi.org/10.1016/j.dam.2004.01.007.

[3] L. Cai, Fixed-parameter tractability of graph modification problems for hereditary properties, Inf. Process. Lett. 58 (4) (1996) 171–176, http://dx.doi.org/10.1016/0020-0190(96)00050-6.

[4] J. Chen, J. Meng, A 2*k* kernel for the cluster editing problem, J. Comput. Syst. Sci. 78 (1) (2012) 211–220, http://dx.doi.org/10.1016/j.jcss.2011.04.001.

[5] F. Protti, M. Dantas da Silva, J.L. Szwarcfiter, Applying modular decomposition to parameterized cluster editing problems, Theory Comput. Syst. 44 (1) (2009) 91–104, http://dx.doi.org/10.1007/s00224-007-9032-7.

[6] P. Damaschke, Cluster editing with locally bounded modifications revisited, in: Combinatorial Algorithms, in: Lecture Notes in Computer Science, vol. 8288, Springer, Heidelberg, 2013, pp. 433–437.

[7] C. Komusiewicz, J. Uhlmann, Cluster editing with locally bounded modifications, Discrete Appl. Math. 160 (15) (2012) 2259–2270, http://dx.doi.org/10.1016/j.dam.2012.05.019.

[8] S. Böcker, A golden ratio parameterized algorithm for cluster editing, J. Discrete Algorithms 16 (2012) 79–89, http://dx.doi.org/10.1016/j.jda.2012.04.005.

[9] S. Böcker, S. Briesemeister, G.W. Klau, Exact algorithms for cluster editing: evaluation and experiments, Algorithmica 60 (2) (2011) 316–334, http://dx.doi.org/10.1007/s00453-009-9339-7.

[10] L. Bastos, L. Satoru Ochi, F. Protti, A. Subramanian, I.C. Martins, R.G.S. Pinheiro, Efficient algorithms for cluster editing, J. Comb. Optim. (2014), http://dx.doi.org/10.1007/s10878-014-9756-7, available online.

[11] P. Damaschke, Fixed-parameter enumerability of cluster editing and related problems, Theory Comput. Syst. 46 (2) (2010) 261–283, http://dx.doi.org/10.1007/s00224-008-9130-1.

[12] M.R. Fellows, J. Guo, C. Komusiewicz, R. Niedermeier, J. Uhlmann, Graph-based data clustering with overlaps, Discrete Optim. 8 (1) (2011) 2–17, http://dx.doi.org/10.1016/j.disopt.2010.09.006.

[13] S. Böcker, J. Baumbach, Cluster editing, in: The Nature of Computation. Logic, Algorithms, Applications, in: Lecture Notes in Computer Science, vol. 7921, Springer, Heidelberg, 2013, pp. 33–44.

[14] B. Mannaa, Cluster editing problem for points on the real line: a polynomial time algorithm, Inf. Process. Lett. 110 (21) (2010) 961–965, http://dx.doi.org/10.1016/j.ipl.2010.08.002.

[15] F.S. Roberts, Indifference graphs, in: Proof Techniques in Graph Theory (Proc. Second Ann Arbor Graph Theory Conf.), Ann Arbor, Mich., 1968, Academic Press, New York, 1969, pp. 139–146.

[16] X. Deng, P. Hell, J. Huang, Linear-time representation algorithms for proper circular-arc graphs and proper interval graphs, SIAM J. Comput. 25 (2) (1996) 390–403, http://dx.doi.org/10.1137/S0097539792269095.