Recognition of dynamic circle graphs

Christophe Crespelle* Emeric Gioan‡ Christophe Paul§

May 28, 2014

Abstract

A *circle graph* is the intersection graph of a set of chords in a circle, this geometric representation of the graph being called a *chord diagram*. A given circle graph may be represented by many different chord diagrams but it is well-known that the set of all possible diagrams can be represented in $O(n)$ space by using the split decomposition. In this paper, we propose an $O(n)$-time algorithm that, given a n-vertex circle graph, maintains the split-decomposition-based representation of a circle graph under vertex deletion and vertex insertion (or asserts that the resulting graph is not circle).

1 Introduction

A *circle graph* is the intersection graph of a set of chords in a circle, also called *chord diagram*: the vertices corresponds to the chords and two vertices are adjacent if and only if their chords intersect. First introduced in the early 1970s as the name *alternance graphs* [7], they have since then been extensively studied, especially in the context of isotropic systems [1, 2] and more recently of rank-width and vertex-minor theory [9]. After a series of polynomial time circle graph recognition algorithm [1, 8], a quadratic algorithm algorithm appeared in mid 1990s [17]. Very recently, an almost linear time algorithm appeared in [11] breaking for the first time the quadratic barrier. Most of these algorithms relies on the split decomposition [6] as circle graphs can be characterized through their split decomposition.

In this paper, we consider a dynamic variant of the recognition problem: given a circle graph G we want to test whether adding a new vertex to G or deleting an existent vertex from G leads to a new circle graph. We provide an $O(n)$-time algorithm to solve this question. The algorithm not only decides whether the modified graph is circle but also maintain a split decomposition based representation of all the possible chord diagrams of the input graph. It can be observed that as long as the split decomposition is maintained, the $O(n)$-time complexity bound is optimal as removing or adding a unique vertex may generate $O(n)$ changes in the split decomposition of the input graph. But still, such a complexity is competitive with quadratic time complexity recognition algorithm of [17]. This problem, known as the dynamic recognition (or representation) problem, has been considered for a large number of graph classes, including subclasses of circle graphs such as distance hereditary graphs and cographs [10, 15] or permutation graphs [5], but also proper interval graphs [13], interval graphs [4], chordal graphs [14] and many others.

*Université Claude Bernard Lyon 1, DANTE/INRIA, LIP UMR CNRS 5668, ENS de Lyon, Université de Lyon - christophe.crespelle@inria.fr

‡LIRMM, CNRS - Université Montpellier 2 - emeric.gioan@lirmm.fr

§LIRMM, CNRS - Université Montpellier 2 - christophe.paul@lirmm.fr

2 Preliminaries

In this paper, we are dealing with connected, simple and loopless graphs. For a graph $G = (V, E)$ we denote $|V| = n$. For a vertex $x \in V$, we denote $N_G(x)$ the neighborhood of x in G. Let S be a subset of vertices of V. The graph *induced* on S is signified by $G[S]$ and the neighborhood of S is $N_G(S) = (\cup_{x \in S} N(x)) \setminus S$. The graph $G + (x, S)$ is obtained from G by adding a new vertex x adjacent to S. We denote by $G - x$ the graph $G[V \setminus \{x\}]$ resulting from the deletion of $x \in V$. A *clique* on n vertices, denoted K_n, is a graph containing an edge between every pair of vertices. A *star* on $n \geqslant 3$ vertices, denoted S_n, is a connected graph containing a vertex, called *centre*, such that every edge is incident to the centre.

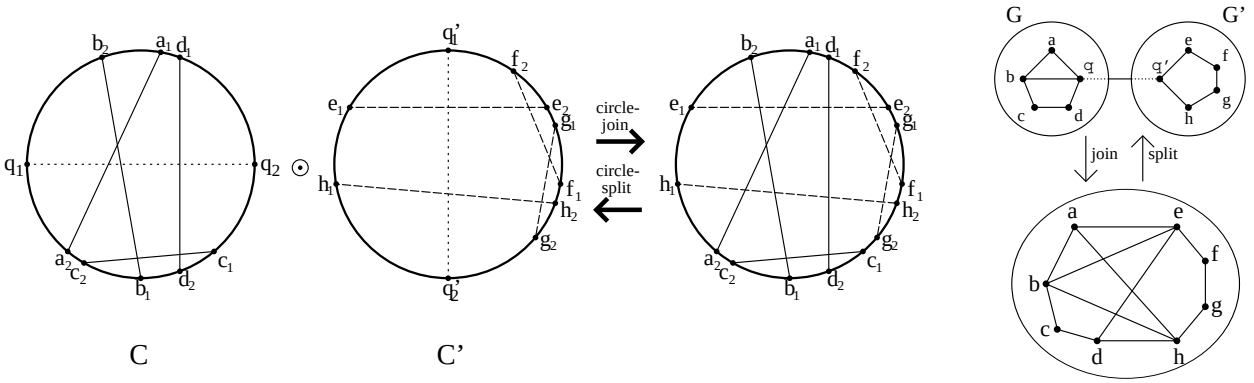

Figure 1: Split and join operations on chord diagrams (left) and GLTs of the same graphs (right).

Split decomposition. A bipartition (A, B) of the vertex set is *trivial* if one of A or B is empty or a singleton. A *split* of a graph $G = (V, E)$ is a non-trivial bipartition (A, B) of V such that every vertex in $A' = N(B)$ is universal to $B' = N(A)$. The sets A' and B' are called the *frontiers* of the split. A graph not containing a split is called *prime*. Cliques and stars are called *degenerate* since every non-trivial bipartition of their vertices is a split. Using the splits, one can recursively decompose an input graph into a set of disjoint graphs $\{G_1, \dots G_k\}$, called *split components*, each of which is either prime or degenerate. There are two cases: if G is prime or degenerate, then return the set $\{G\}$; if G is neither prime nor degenerate, it contains a split (A, B), with frontiers A' and B'. The split components of G is then the union of the split components of the graphs $G[A] + (a, A')$ and $G[B] + (b, B')$, where a and b are new vertices, called *markers*. Observe that the split decomposition process naturally defines a decomposition tree whose nodes are the split components. It can be represented as the so-called *skeleton graph* (see [3] for example). Here, we follow the framework of *graph-labelled trees* introduced in [10, 12]:

A *graph-labelled tree* (GLT) is a pair $(T, \mathcal{F})$, where T is a tree and $\mathcal{F}$ a set of graphs, such that each node u of T is *labelled* by the graph $G(u) \in \mathcal{F}$, and there exists a bijection ρ_u between the edges of T incident to u and the vertices of $G(u)$. The vertices of $G(u)$ are called *marker vertices*, and the edges between marker vertices in $G(u)$ are called *label-edges*. We will abusively consider a leaf ℓ of T as a marker vertex (of a single vertex labelled-graph). The marker vertices $\rho_u(e)$ and $\rho_v(e)$ of the tree-edge $e = uv$ are called the *extremities* of e. Let $(T, \mathcal{F})$ be a GLT. The marker vertices q and q' are *accessible* from one another if there is a sequence Π of marker vertices such that every two consecutive elements of Π are either the vertices of a label-edge or the extremities of a tree-edge; and the edges thus defined alternate between tree-edges and label-edges. The *accessibility graph* of $(T, \mathcal{F})$, denoted $Gr(T, \mathcal{F})$, is the graph whose vertices are the leaves of T, with an edge between

two distinct leaves ℓ and ℓ' if and only if they are accessible from one another. Conversely, we may say that $(T, \mathcal{F})$ *is a GLT of* $Gr(T, \mathcal{F})$.

Let us observe that every tree edge e of a GLT $(T, \mathcal{F})$ defines a split (A, B) of the accessibility graph $Gr(T, \mathcal{F})$ where A and B respectively contains the vertices corresponding to the leaves of the two connected components of $T - e$. Naturally a graph-label in a GLT containing a split can be transformed into two adjacent graph-labels (split operation), and conversely (join operation), preserving the accessibility graph of the GLT, see Figure 1. Cunningham [6] showed that every graph admits a canonical split decomposition tree. In term of GLT's, this translates as follows:

Theorem 2.1. *[6, 10, 12] Let G be a connected graph. There exist a unique GLT $(T, \mathcal{F})$ whose labels are either prime or degenerate, having a minimal number of nodes and such that $G = Gr(T, \mathcal{F})$. This GLT is called the* split-tree *of G and denoted $ST(G)$.*

Circle graphs. It is well-known that a graph is a circle graph if and only if all its split components (graph-labels of its GLT) are circle graphs [1]. As a consequence, because cliques and stars are circle graphs, the recognition of circle graphs can be reduced to the recognition of prime circle graphs. That was the approach developed in [16, 17]. Every circle graph $G = (V, E)$ has a representation as a chord diagram, but such a representation is not unique. Indeed, as witness by examples of cliques, a circle graph may have exponentially many distinct chord diagrams. However, it is known that a prime circle graph has a unique chord diagram (up to mirror) [1, 3]. We call *augmented split tree* of a circle graph its split tree together with a chord diagram of its labels. Figure 1 illustrates split and join operations on chord diagrams.

3 Our result

Our original result is stated in the following theorem.

Theorem 3.1. *Given the augmented split tree $ST(G)$ of a connected circle graph $G = (V, E)$, in $O(n)$-time, we can :*

- *compute the augmented split trees of every connected component of $G - x$;*
- *decide if $G' = G + (x, S)$, with $S \subseteq V$, is circle and if so compute its augmented split tree.*

Below we provide a brief description of the key ingredients of its proof. The construction consists in dealing with prime circle graphs for vertex insertion and deletion, and incorporating this case in the general dynamic construction from [12, Section 4], which provides a combinatorial description of how the split tree $ST(G + (x, S))$ is obtained from the split tree $ST(G)$, and conversely, with S being a subset of the vertices of G. Technically, the specificity of prime circle graphs in terms of split tree updates is given by the following crucial lemma.

Lemma 3.2. *Let x be a vertex of a prime circle graph $G = (V, E)$. The split tree $ST(G - x)$ is a caterpillar and every pair (A, B) and (C, D) of splits of $G - x$ is nested, that is $A \subset C$ or $C \subset A$.*

Vertex deletion. Removing a vertex x from a circle graph G always yields a new circle graph $G' = G - x$. Removing the leaf ℓ_x and its opposite marker vertex q_u in $G(u)$ (where u is the node adjacent to ℓ_x) results in a GLT of $G - x$ from which we can compute $ST(G - x)$. The difficult case is when $G(u)$ is a prime graph because then the removal of q_u may generate new splits to be identified in $O(n)$-time. Computing $ST(G - x)$ is performed by searching the chord diagram of G in order to identify in $O(n)$-time a maximal set of nested splits.

Vertex insertion. As for the deletion, the difficult case for vertex insertion is when $G + (x, S)$ is a prime graph. This case can be detected by a simple $O(n)$-time marking process of $ST(G)$ with respect to S described in [12]. The previous lemma is also useful in that situation, as it implies that for $G + (x, S)$ to be a circle graph, the subtree $T(S)$ of $ST(G)$ spanned by the leaves corresponding to S has to be a caterpillar. If this is not the case, then insertion algorithm can directly stop. Let us assume that $T(S)$ is indeed a caterpillar. Basically, the insertion algorithm amounts to perform a series of join operations to shrink $ST(G)$ in a single node GLT and then insert in the associated chord diagram $\mathcal{C}$ a new chord c_x representing vertex x. As a join operation can be implemented using chord diagrams, $\mathcal{C}$ results from the series or performed join operations. Those join operations, maintaining chord diagrams as well, can be done in linear time. As the total size of the augmented $ST(G)$ in linear in $|V|$, it follows that shrinking $ST(G)$ into a single node GLT requires $O(n)$-time. Now observe that as $G + (x, S)$ is prime, it has a unique chord diagram. It follows that if $G + (x, S)$ is a circle graph, there is a unique way to insert c_x in $\mathcal{C}$. Again testing the insertion can be achieved in $O(n)$ time.

References

[1] A. Bouchet. Reducing prime graphs and recognizing circle graphs. *Combinatorica*, 7:243–254, 1987.

[2] A. Bouchet. Circle graph obstructions. *Journal of Combinatorial Theory Series B*, 60:107–144, 1994.

[3] B. Courcelle. Circle graphs and monadic second-order logic. *Journal of Applied Logic*, 6:416–442, 2008.

[4] C. Crespelle. Fully dynamic representation of interval graphs. In *International Workshop on Graph Theoretical Concepts in Computer Science (WG)*, 2009.

[5] C. Crespelle and C. Paul. Fully dynamic algorithm for recognition and modular decomposition of permutation graphs. *Algorithmica*, 58(2):405–432, 2010.

[6] W.H. Cunningham and J. Edmonds. A combinatorial decomposition theory. *Canadian Journal of Mathematics*, 32(3):734–765, 1980.

[7] S. Even and A. Itai. Queues, stacks and graphs. *Theory of Machines and Computations*, pages 71–86, 1971.

[8] C.P. Gabor, W.L. Hsu, and K.J. Suppovit. Recognizing circle graphs in polynomial time. *Journal of ACM*, 36:435–473, 1989.

[9] J. Geelen and S.I. Oum. Circle graph obstructions under pivoting. *Journal of Graph Theory*, 2008.

[10] E. Gioan and C. Paul. Split decomposition and graph-labelled trees: Characterizations and fully dynamic algorithms for totally decomposable graphs. *Discrete Applied Mathematics*, 160(6):708–733, 2012.

[11] E. Gioan, C. Paul, M. Tedder, and D. Corneil. Practical and efficient circle graph recognition. *Algorithmica*, 2013.

[12] E. Gioan, C. Paul, M. Tedder, and D. Corneil. Practical and efficient split-decomposition via graph-labelled trees. *Algorithmica*, 2013.

[13] P. Hell, R. Shamir, and R. Sharan. A fully dynamic algorithm for recognizing and representing proper interval graphs. *SIAM Journal on Computing*, 31(1):289–305, 2002.

[14] L. Ibarra. Fully dynamic algorithms for chordal graphs. In *Annual ACM-SIAM symposium on Discrete algorithms (SODA)*, pages 923–924, 1999.

[15] R. Shamir and R. Sharan. A fully dynamic algorithm for modular decomposition and recognition of cographs. *Discrete Applied Mathematics*, 136(2-3):329–340, 2004.

[16] J. Spinrad. Prime testing for the split decomposition of a graph. *SIAM Journal on Discrete Mathematics*, 2(4):590–599, 1989.

[17] J. Spinrad. Recognition of circle graphs. *Journal of Algorithms*, 16:264–282, 1994.