数理解析研究所講究録
第1644巻 2009年 128-134

Reconstruction of Connected Interval Graphs

Masashi Kiyomi, Toshiki Saitoh, and Ryuhei Uehara
School of Information Science, JAIST,
Asahidai 1-1, Nomi, Ishikawa 923-1292, Japan.
{mkiyomi,toshikis,uehara}@jaist.ac.jp

Abstract

The graph reconstruction conjecture is a long-standing open problem in graph theory. There are many algorithmic studies related it besides mathematical studies, such as DECK CHECKING, LEGITIMATE DECK, PREIMAGE CONSTRUCTION, and PREIMAGE COUNTING. We study these algorithmic problems limiting the graph class to interval graphs. Since we can solve GRAPH ISOMORPHISM for interval graphs in polynomial time, DECK CHECKING for interval graphs is easily done in polynomial time. We present in this paper that the other three problems above are solvable in polynomial time on connected interval graphs.

Keywords: the graph reconstruction conjecture, interval graphs, polynomial time algorithm

1 Introduction

Given a simple graph $G = (V, E)$, we call the multi-set $\{G - v \mid v \in V\}$ the deck of G where $G - v$ is a graph obtained from G by removing vertex v and incident edges. The graph reconstruction conjecture by Uram and Kelly[1] is that for any multi-set D of graphs with at least two vertices there is at most one graph whose deck is D. We call a graph whose deck is D a preimage of D. No counter example is known for this conjecture, and there are many mathematical results about this conjecture. For example trees, regular graphs, and disconnected graphs are reconstructible (i.e. the conjecture is true for these classes) [5]. About interval graphs, Rimscha showed that interval graphs are recognizable in the sense that looking at the deck of G one can decide whether or not G belongs to interval graphs [10]. Rimscha also showed in the same paper that many subclasses of perfect graphs including perfect graphs themselves are recognizable, and some of subclasses including unit interval graphs are reconstructible. There are many good surveys about this conjecture. See for example [1, 4].

Besides these mathematical results, there are some algorithmic results. We enumerate the algorithmic problems that we address in this paper.

- Given a graph G and a multi-set of graphs D, check whether D is a deck of G (DECK CHECKING).
- Given a multi-set of graphs D, determine whether there is a graph whose deck is D (LEGITIMATE DECK).
- Given a multi-set of graphs D, construct a graph whose deck is D (PREIMAGE CONSTRUCTION).
- Given a multi-set of graphs D, compute the number of (pairwise nonisomorphic) graphs whose decks are D (PREIMAGE COUNTING).

[1] Determining the first person who proposed the graph reconstruction conjecture is difficult, actually. See [4] for the detail.

Graph class specified versions of LEGITIMATE DECK and PREIMAGE CONSTRUCTION assume that preimages are in the specified graph class. Kratsch and Hemaspaandra showed that these problems are solvable in polynomial time for graphs of bounded degree, partial k-trees for any fixed k, and graphs of bounded genus, in particular for planner graphs [7]. In the same paper they proved many GI related complexity results.

There is a linear time algorithm for determining if given two interval graphs are isomorphic [9]. Thus developing a polynomial time algorithm for DECK CHECKING for interval graphs is easy.

Theorem 1 *There is an $O(n(n+m))$ time algorithm of DECK CHECKING for n-vertex m-edge graph and its deck (or a deck candidate) that consists of interval graphs.*

We will give the proof in Section 3.

LEGITIMATE DECK, PREIMAGE CONSTRUCTION, and PREIMAGE COUNTING for connected interval graphs are solvable by almost the same algorithm. In order to develop such an algorithm we show that given a set of n interval graphs D there is at most $O(n^2)$ connected interval graphs (preimages) whose deck is D. Further we can construct such $O(n^2)$ preimage candidates. Our algorithm checks these $O(n^2)$ candidates one by one whether its deck is D with DECK CHECKING algorithm. Our algorithm constructs n preimage candidates from $O(n)$ different interval representations of each interval graph in D by inserting an interval to them. The key is that the number of preimage candidates is $O(n^2)$ while a naive algorithm which inserts an interval to an interval representation constructs $\Omega(2^n)$ candidates (Consider the case that $O(n)$ intervals terminate at some point t, and we insert a new left end-point to t. The number of the ways of insertions is $\Omega(2^n)$ since there are $O(n)$ times choices whether the new interval intersects the old ones. Further, there may be many, say $O(2^n)$, different compact interval representations for an interval graph. Therefore the number of preimage candidates will be very huge if we construct the candidates from all of them).

The following is our main theorem.

Theorem 2 *There are $O(n^3(n+m))$ time algorithms for LEGITIMATE DECK and PREIMAGE CONSTRUCTION, and there is an $O(n^4(n+m))$ time algorithm for PREIMAGE COUNTING, for connected interval graphs.*

Note that n is the number of vertices and m is the number of edges in the preimage. Kelly's lemma [5] shows that we can compute n and m from the deck.

We state terminologies in Section 2, then explain about interval graphs in Section 3. In Section 3 we introduce many small lemmas for those who unfamiliar to interval graphs. Most of these lemmas may be well-known and/or basic for those who familiar to interval graphs and the theory of PQ-tree [2] and MPQ-tree [6]. However these lemmas play important roles in this paper. Then we show that the number of preimage candidates is $O(n^2)$, and we present our algorithm in Section 4. Finally we make some remarks in Section 5.

2 Terminology

Graphs in this parer are all simple and undirected, unless explicitly stated. We denote by $N_G[v]$ the closed neighbor set of vertex v in graph G. "Closed" means that $N_G[v]$ contains v itself. We denote by $\deg_G(v)$ the degree of vertex v in graph G. We omit the subscript G when there is no confusion about the base graph. The sum of degrees of all vertices in graph G is denoted by degsum(G). Notice that degsum(G) is equal to twice the number of edges in G. We denote by $\tilde{G}$ the graph obtained by adding one universal vertex to the graph G such that the vertex connects to every vertex in G. Notice that $\tilde{G}$ is always connected. Given two graphs G_1 and G_2, we define the disjoint union $G_1 \dot{\cup} G_2$ of G_1 and G_2 as $(V_1 \dot{\cup} V_2, E_1 \dot{\cup} E_2)$ such that (V_1, E_1) is isomorphic to G_1, and (V_2, E_2) is isomorphic to G_2, where $\dot{\cup}$ means the disjoint union.

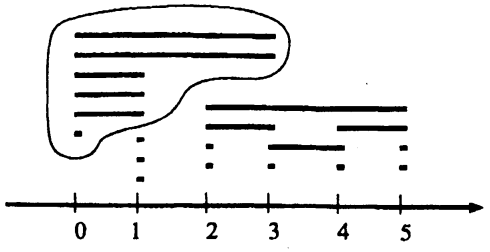

Figure 1: The figure shows a compact interval representation of an interval graph. Every interval graph has at least one compact interval representation. Vertices corresponding to the enclosed intervals are end-vertex set.

3 Interval Graphs

A graph $G = (V, E)$ with $V = \{v_1, v_2, \ldots, v_n\}$ is an interval graph iff there is a multi-set $\mathcal{I} = \{I_{v_1}, I_{v_2}, \ldots, I_{v_n}\}$ of closed intervals on the real line such that $\{v_i, v_j\} \in E$ if and only if $I_{v_i} \cap I_{v_j} \neq \emptyset$ for each i and j with $1 \leq i, j \leq n$. We call the multi-set $\mathcal{I}$ an *interval representation* of G. An interval graph may have infinitely many interval representation. We use tractable one called compact interval representation among them.

3.1 compact representation and basic lemmas

Definition 1 *An interval representation $\mathcal{I}$ of an interval graph $G = (V, E)$ is compact iff*

- *coordinates of end-points of intervals in $\mathcal{I}$ are finite non-negative integers (We denote by K the largest coordinates of end-points for convenience. We sometimes call K the length of $\mathcal{I}$),*
- *there exists at least one end-point whose coordinate is k for every integer $k \in [0, K]$, and*
- *interval multi-set $\mathcal{I}_k = \{I \in \mathcal{I} \mid k \in I\}$ differs from $\mathcal{I}_l = \{I \in \mathcal{I} \mid l \in I\}$, and they do not include each other, for every distinct integers $k, l \in [0, K]$.*

We show an example of a compact interval representation of an interval graph in Fig. 1. Note that there may be still multiple compact interval representations of an interval graph. However compact interval representations have some good properties.

Lemma 1 *Let $\mathcal{I}$ and $\mathcal{J}$ be compact interval representations of an interval graph $G = (V, E)$, and let K_1 be the length of $\mathcal{I}$, and let K_2 be the length of $\mathcal{J}$. Then the following holds.*

$$\{\{I \in \mathcal{I} \mid 0 \in I\}, \{I \in \mathcal{I} \mid 1 \in I\}, \ldots, \{I \in \mathcal{I} \mid K_1 \in I\}\} =$$
$$\{\{I \in \mathcal{J} \mid 0 \in I\}, \{I \in \mathcal{J} \mid 1 \in I\}, \ldots, \{I \in \mathcal{J} \mid K_2 \in I\}\}$$

Proof: We denote by $\bar{\mathcal{I}}$ the set of multi-set of intervals $\{\{I \in \mathcal{I} \mid 0 \in I\}, \{I \in \mathcal{I} \mid 1 \in I\}, \ldots, \{I \in \mathcal{I} \mid K_1 \in I\}\}$, and we denote by $\bar{\mathcal{J}}$ the set of multi-set of intervals $\{\{I \in \mathcal{J} \mid 0 \in I\}, \{I \in \mathcal{J} \mid 1 \in I\}, \ldots, \{I \in \mathcal{J} \mid K_2 \in I\}\}$. The vertices represented by the multi-set of intervals $\mathcal{I}_i = \{I \in \mathcal{I} \mid i \in I\}$ correspond to a clique in G. Assume that $\mathcal{I}_i$ never appears in $\bar{\mathcal{J}}$ for some i. Since $\mathcal{I}_i$ represents a clique C, there must be a set of intervals representing a clique C' containing C in $\bar{\mathcal{J}}$ (otherwise, clique C can not be represented in $\mathcal{J}$). Then for the same reason, $\bar{\mathcal{I}}$ must contain a set of intervals representing a clique containing C'. This contradicts the compactness of $\mathcal{I}$. □

From the proof of Lemma 1, the following lemmas are straight-forward.

Lemma 2 *Let $\mathcal{I}$ be a compact interval representation of an interval graph $G = (V, E)$, and let K be the length of $\mathcal{I}$. Then $\{I \in \mathcal{I} \mid i \in I\}$ for each $i \in \{0, \ldots, K\}$ corresponds to each maximal clique of G.*

boolean function *deck-checking*(**graph** $G = (V, E)$) {
  Let G' be an empty graph.
  for each vertex $v \in V$ { $G' := G' \dot{\cup} (G - v)$. }
  if G' is isomorphic to $G_1 \dot{\cup} G_2 \dot{\cup} \ldots \dot{\cup} G_n$
    return True **else** **return False.**
}

Figure 2: The deck checking algorithm.

Lemma 3 *The length of a compact interval representation of an n-vertex interval graph is at most n.*

Note that the number of maximal cliques in an n-vertex interval graph is at most n (see [3]).

Lemma 4 *All the compact interval representations of an interval graph have the same length.*

Lemma 5 *Intervals in different compact interval representations corresponding to an identical vertex have the same length.*

From Lemma 5, lengths of intervals corresponding to a vertex that corresponds to an interval of length zero in some interval representation are always (i.e. in any interval representations) zero. Such vertex is called *simplicial.*

Notice that all the members in a deck of an interval graph are interval graphs since removing a vertex from an interval graph results in another interval graph.

3.2 deck checking

Now we prove Theorem 1. Our main algorithm enumerates the preimage candidates, and checks whether each candidate is really a preimage of the input deck. Thus the theorem is one of the basic part of our algorithm.

Proof: [Proof of Theorem 1] We can determine whether or not the given multi-set $D = \{G_1, G_2, \ldots, G_n\}$ is a deck of the input graph G by checking whether or not $G_1 \dot{\cup} G_2 \dot{\cup} \ldots \dot{\cup} G_n$ is isomorphic to the deck of G. Since the disjoint union of two interval graphs is an interval graph, we can use well-known linear time isomorphism algorithm [9] for this checking. We describe the algorithm in Fig. 2. Since the number of vertices of $G_1 \dot{\cup} \ldots \dot{\cup} G_n$ is $O(n^2)$, and since the number of edges of $G_1 \dot{\cup} \ldots \dot{\cup} G_n$ is $O(mn)$, the time complexity of this algorithm is $O(n(n+m))$. □

4 Main Algorithm

First we define end-vertex set. The end-vertex set is intuitively a set of vertices whose corresponding intervals are at the left end. Our algorithm adds a vertex adjacent to all the vertices in an end-vertex set of an interval graph in the input deck. This enables us to avoid exponential times' constructions of preimage candidates.

Definition 2 *For an interval graph $G = (V, E)$, we call a vertex subset $S \subset V$ an end-vertex set iff in some compact interval representation of G all the coordinates of the left end-points of intervals corresponding to vertices in S are 0, and S is maximal among such vertex subsets.*

See Fig. 1 for example. It is well-known that an end-vertex set has at least one simplicial vertex.

We show some simple lemmas about end-vertex sets. We can estimate that the number of essentially different preimage candidates is $O(n^2)$ by these lemmas.

Lemma 6 *Let S be end-vertex set of an interval graph $G = (V, E)$. If two vertices v and w in S have the same degree, then $N[v]$ and $N[w]$ are the same vertex subset of G.*

Proof: The statement is clear from the definition of compact interval representation (see Fig. 1 for the better understanding). □

Lemma 7 *A connected interval graph has at most $O(n)$ end-vertex sets.*

Proof: An end-vertex set of an interval graph G is in the form $\{I \in \mathcal{I} \mid 0 \in I\}$ for some interval representation $\mathcal{I}$ of G. Thus, from Lemma 1 and 3, there are at most $O(n)$ end-vertex sets for G. □

Now we refer the well-known lemma about the degree sequence.

Lemma 8 (Kelly's Lemma [5]) *We can calculate the degree sequence of a preimage of the input n graphs in $O(n)$ time, if we know the number of edges in each input graph.*

Proof: Let $G_1, G_2, \ldots, G_n$ be the input graphs. Assume that graph G has a deck $\{G_1, G_2, \ldots, G_n\}$. Then there are vertices $v_1, v_2, \ldots, v_n$ such that G_i is obtained by removing v_i from G for each i in $\{1, 2, \ldots, n\}$. Thus

$$\mathrm{degsum}(G_i) = \mathrm{degsum}(G) - 2\mathrm{deg}_G(v_i)$$

holds for each $i \in \{1, 2, \ldots, n\}$. Hence we have

$$\mathrm{degsum}(G) = \frac{\sum_{i=1}^{n} \mathrm{degsum}(G_i)}{n-2}.$$

Therefore we can easily calculate the degree sequence of G, i.e., $(\mathrm{degsum}(G) - \mathrm{degsum}(G_1))/2$, $(\mathrm{degsum}(G) - \mathrm{degsum}(G_2))/2, \ldots, (\mathrm{degsum}(G) - \mathrm{degsum}(G_n))/2$.

We can calculate $\mathrm{degsum}(G_i)$ in constant time, provided we know the number m_i of edges in G_i, for $\mathrm{degsum}(G_i)$ is equal to $2m_i$. Thus the time complexity to calculate $\mathrm{degsum}(G)$ is $O(n)$, and the total time complexity to obtain the degree sequence of G is also $O(n)$. □

Now we present an algorithm for reconstructing a connected interval graph. Suppose that an n-vertex connected interval graph G has a deck of interval graphs $\{G_1, G_2, \ldots, G_n\}$. Let $\mathcal{I}$ be a compact interval representation of G. There must be an index $i \in \{1, \ldots, n\}$ such that G_i is obtained by removing a simplicial vertex s in the end-vertex set S corresponding to $\mathcal{I}$. We want to reconstruct G from $G_1, \ldots, G_n$. To do so, we first show that we can reconstruct G if we know the index i. Once we prove this, we can reconstruct G by checking if G_j is the desired G_i for every $j \in \{1, \ldots, n\}$.

It is clear that $S \setminus \{s\}$ is contained in some end-vertex set S' of G_i. Of course we do not know S' if we do not know $S \setminus \{s\}$. However the number of the candidates of S' is $O(n)$ by Lemma 7. Thus checking if each candidate is S' can be done by $O(n)$ times trying of the algorithm below. Let $\mathcal{I}_i$ be an interval representation of G_i whose corresponding end-vertex set is S'. Notice that $\mathcal{I}_i$ is easily obtained in $O(n+m)$ time by using the data structure called MPQ-tree [6] if we know S'.

Now we try to know $S \setminus \{s\}$. If we know $S \setminus \{s\}$, we can obtain G, since G has the interval representation obtained from $\mathcal{I}_i$ by extending intervals corresponding to vertices in $S \setminus \{s\}$ to the left by one and adding an interval $[-1, -1]$. Therefore we need to know $S \setminus \{s\}$. Since we know the degree sequence of G_i, and we can know the degree sequence of G by Lemma 8, we can know the degree sequence of $S \setminus \{s\}$. We denote the degree sequence by $(d_1, d_2, \ldots, d_l)$. Now we can obtain $S \setminus \{s\}$; $S \setminus \{s\}$ is the subset of S' such that whose degree sequence in G_i is $(d_1 - 1, d_2 - 1, \ldots, d_l - 1)$. Notice that there may be many subsets of S' whose degree sequences in G_i are $(d_1 - 1, d_2 - 1, \ldots, d_l - 1)$. However Lemma 6 guarantees that any of such subsets can be $S \setminus \{s\}$, i.e. all the graphs reconstructed in the assumption that some subset of S' whose degree sequence is $(d_1 - 1, d_2 - 1, \ldots, d_l - 1)$ are $S \setminus \{s\}$ are isomorphic to each other. Therefore we can reconstruct G. The whole algorithm is described in Fig 3.

```
for each G_i(i = 1, 2, ..., n) {
   for each end-vertex set S' of G_i {

      Let I be an interval representation of G_i
       whose corresponding end-vertex set is S'.

      Compute the degree sequence (d_1, ..., d_l) of S \ {s}.
      Let S'' be a subset of S' whose degree sequence in G_i is (d_1 - 1, ..., d_l - 1).
      Let G be an interval graph
       whose interval representation is obtained from I
       by extending interval corresponding to vertices in S'' to the left by one
       and adding an interval [-1, -1].

      if deck-checking(G) = True
         output G.
   }
}
return No if the algorithm has output no graph.
```

Figure 3: The algorithm for reconstructing connected interval graphs.

Now we consider the time complexity of this algorithm. Because of the space limitation we omit the detail of the basic algorithms about MPQ-tree. For each G_i, calculating an MPQ-tree of G_i in $O(n+m)$ time helps us to list each S' and I in $O(n)$ time. Computing the degree sequence $(d_1, d_2, \ldots, d_l)$ takes $O(n)$ time from Lemma 8. Since obtaining S'' needs sorting of the degree sequence, it requires $O(n \log n)$ time. It is clear that reconstructing an interval graph from its interval representation takes $O(n+m)$ time, if the end-points of intervals are sorted. DECK CHECKING algorithm costs $O(n(n+m))$. Therefore the total time complexity of this algorithm is $O(n((m+n)+n(n+m+n\log n+n(n+m)))) = O(n^3(m+n))$. Note that we have to check every output preimage is not isomorphic to each other for PREIMAGE COUNTING. Since the number of output preimage may be $O(n^2)$, we need $O(n^4(n+m))$ time for this checking. If the graph reconstruction conjecture is true, the time complexity of this checking can be omitted.

Theorem 3 *There is a polynomial time algorithm that lists up connected interval graphs that are preimages of the input n interval graphs. The time complexity for outputting one connected interval graph is $O(n^3(n+m))$, and that for outputting all is $O(n^4(m+n))$.*

Therefore we have the main theorem (Theorem 2).

5 Concluding Remarks

The algorithms we described does not help directly the proof of the graph reconstruction conjecture on interval graphs. The conjecture on interval graphs remains to be open.

The complexities of graph reconstruction problems are strongly related to that of graph isomorphism problems. To develop a polynomial time algorithm for a graph reconstruction problem restricting inputs to be in GI-hard graph class seems very hard. Since graph isomorphisms of circular-arc graphs are polynomial time solvable, circular-arc graph reconstruction problem may be a good challenge.

References

[1] J. A. Bondy: A graph reconstructor's manual. *Surveys in Combinatorics, London Mathematical Society Lecture Note Series*, vol. 166 (1991) 221–252.

[2] K. S. Booth and G. S. Lueker: Testing for the consecutive ones property, interval graphs, and graph planarity using *PQ*-tree algorithms. *Journal of Computer and System Sciences*, vol. 13 (1976) 335–379.

[3] D. R. Fulkerson and O. A. Gross: Incidence matrices and interval graphs. *Pacific Journal of Mathematics*, vol. 15 (1965) 835–855.

[4] F. Harary: A survey of the reconstruction conjecture. Graphs and Combinatorics, *Lecture Notes in Mathematics*, vol. 406 (1974) 18–28.

[5] P. J. Kelly: A congruence theorem for trees. *Pacific Journal of Mathematics*, vol. 7 (1957) 961–968.

[6] N. Korte and R. H. Möhring: An incremental linear-time algorithm for recognizing interval graphs. *SIAM Journal on Computing*, vol. 18 (1989) 68–81.

[7] D. Kratsch and L. A. Hemaspaandra: On the complexity of graph reconstruction, *Mathematical Systems Theory*, vol. 27 (1994) 257–273.

[8] C. G. Leckerkerker and J. Ch. Boland: Representation of a finite graph by a set of intervals on the real line. *Fundamenta Mathematicae*, vol. 51 (1962) 45–64.

[9] G. S. Lueker and K. S. Booth: A linear time algorithm for deciding interval graph isomorphism. *Journal of the ACM*, vol. 26 (1979) 183–195.

[10] M. von Rimscha: Reconstructibility and perfect graphs. *Discrete Mathematics*, vol. 47 (1983) 283–291.