CS762: Graph-Theoretic Algorithms
Lecture 3: Algorithms for interval graphs
January 11, 2002

Scribe: Henning Stehr

Abstract

In this lecture we define an algorithm for calculating the chromatic number and the maximum clique number on a superset of interval graphs. To characterize this superset we introduce the notion of a perfect elimination order. We show that both problems, which are NP-hard in general, can be solved in linear time on this class.

1 Introduction

In the previous lecture we have defined the class of interval graphs. Now we would like to address the question, if certain known to be hard problems can be solved efficiently on interval graphs. The two problems that we will deal with are graph-coloring and computing the size of a maximum clique.

Graph coloring is one of the classical problems in graph theory and has been widely studied. It occurs in a number of applications such as assignment and scheduling problems (see [Wer85]). It is currently being used in algorithms for register allocation (see [CH90]). Especially the coloring of planar graphs, known as map-coloring, has been studied intensively. Probably the most famous result is Appel and Haken's proof that every planar graph is 4-colorable [AH76].

Just like coloring, the finding of maximum cliques has numerous applications in such diverse fields as operations research, phylogenetic analysis and pettern recognition. For a sample applications of the latter see [Oga86].

In this paper we will show that both of these problems can be solved efficiently and with the same algorithm on interval graphs. Moreover we will introduce a property of graphs that allows the algorithms to be optimal. In upcoming lectures we will intensively study this class and see that it is the class of chordal graphs.

Where no further reference is cited, the material for this lecture can be found in [Gol80], Sec. 4.7. The definition for the problems are given in the sections of this paper. All we need to start from is the definition of interval graphs:

Definition 1 ***(Interval Graph)*** *Given a set of intervals, a graph with each of the vertices corresponding to one of the intervals and two vertices connected by an edge if and only if the two intervals intersect is called an interval graph.*

2 Graph coloring

In this section we will study the problem of vertex coloring on interval graphs, i.e. an assignment of a set of colors to the vertices of the graph such that for every edge the two endpoint do not have the same color. We will not consider edge coloring here.

Definition 2 *The Chromatic Number of a graph G, denoted $\chi(G)$ is the minimum number of colors needed to color G.*

Definition 3 *As k-coloring we denote the decision problem: Is it possible to color a graph with k colors?*

2-coloring would be the same as to ask: Is the graph bipartite? As for the satisfiability problem, where 2-SAT is polynomial ([GJ79], p.50) and 3-SAT is NP-hard it can be shown that 2-coloring is linear and k-coloring for $k \geq 3$ is NP-hard. We will give an outline of the proof for 3-coloring.

Theorem 1 *3-coloring is NP-hard.*

Note that most of the problems we deal with in this course will be in NP. However we will focus on proving NP-hardness only.

Proof: We show that NAE-3-SAT which is known to be NP-hard (see [GJ79], p.259) reduces to 3-coloring. Given an instance of NAE-3-SAT we will construct a graph and show that if the instance has a solution the graph is 3-colorable.

Definition 4 *NAE-3-SAT: Given a set U of variables and a collection C of clauses over U such that each clause $c \in C$ has $|c| = 3$. Is there a truth assignment for U such that each clause in C has at least one true literal and at least one false literal?*

We construct a graph G as illustrated in Figure 1 :

- For each variable in $u \in U$ we define two vertices corresponding to u and $\bar{u}$ and three edges that connect u with $\bar{u}$ and both with a common central vertex r.
- For each clause $c \in C$ we define a triangle for the three literals and connect each of the three vertices to the corresponding vertex from above.
- For the coloring we use $\{R, T, F\}$ where R is the color of the central vertex r.
- Hence for each variable, one of u and $\bar{u}$ must be assigned *true* (color T), the other *false* (color F).

What remains to be shown is that the solutions for the NAE-3-SAT are exactly the 3-colorings of G.
'$\Longrightarrow$' Given a NAE-3-SAT truth assignment we set the inner vertices corresponding and every triangle will be colorable. There will be at most twice the same color so we can use R as a replacement and obtain three different colors.
'$\Longleftarrow$' If the graph is 3-colorable, we can set the inner parts to be opposite to the solution in the triangle.

We now have shown that if we can solve 3-coloring then we can solve NAE-3-SAT. Hence, under the assumption that NAE-3-SAT is NP-complete, so is 3-coloring.

$\square$

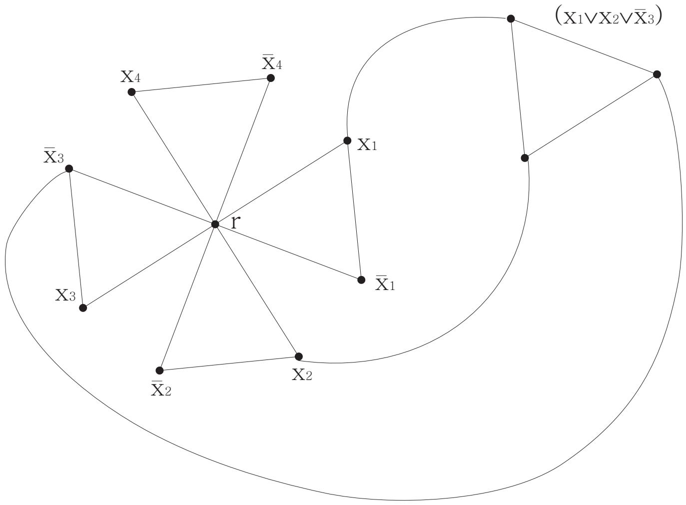

Figure 1: Proof of the NP-completeness of 3 coloring: The graph is 3-colorable if and only if there exists a truth assignment for the corresponding instance of NAE-3-SAT.

3 A greedy algorithm for coloring graphs

While finding the chromatic number of a graph G is NP-complete in general, we can use a greedy algorithm to find a coloring (with probably more than $\chi(G)$ colors).

1. `INPUT: Graph` G`, vertex ordering` $v_1, \ldots, v_n$
2. `for` i `=` $1 \ldots n$ `{`
3. `let` j `be the smallest color not used in Pred(`v_i`).`
4. `color` v_i `with color` j`.`
5. `}`

This algorithm will find a coloring of G with at most $\max_i \{\text{indeg}(v_i) + 1\}$ colors (which can be very bad). Note that the result highly depends on the given vertex order.

Remark 1 *The above statement implies that we can do 6-coloring for planar graphs. However, the famous result of Appel and Haken [AH76] shows that any planar graph is 4-colorable.*

4 Optimality conditions for the coloring algorithm

We will now investigate under what conditions the greedy algorithm is optimal, i.e. will find the minimum number of colors.

Theorem 2 *For an interval graph, using the natural vertex order, the greedy algorithm is optimal.*

Proof: We know that our algorithm finds a coloring with at most

$$\max_i \{\text{indeg}(v_i) + 1\}$$

colors. Let i^* be the index where this maximum is achieved. We know from lecture 1 that the predecessors of a vertex in an interval graph form a clique. So, $\text{Pred}(v_i^*) \cup v_i^*$ is a clique of size $\text{indeg}(v_i^*)+1$. To color this clique we need at least $\text{indeg}(v_i^*)+1$ colors.

Since the greedy algorithm uses exactly this number of colors, that we have just shown to be the minimum number, it is optimal.

□

The proof shows that all we really need, to obtain optimality is a special vertex order, so we give this property a name:

Definition 5 *A vertex order $\{v_1 \dots v_n\}$ is called a* ***perfect elimination order*** *if Pred(v_i) is a clique for every $i \in \{1 \dots n\}$.*

So what we really showed with the previous proof was:

Theorem 3 *If a graph G has a perfect elimination order then we can compute $\chi(G)$ in linear time.*

Remark 2 *Even if we know (or suspect) a graph to have a perfect elimination order, we still have to find it. We will show later that this can be done in linear time (e.g. with LexBFS). For details see the notes for lecture 5 or [Gol80], Sec. 4.3.*

Remark 3 *A graph having a perfect elimination order does not necessarily have to be an interval graph. Figure 2 shows a counterexample.*

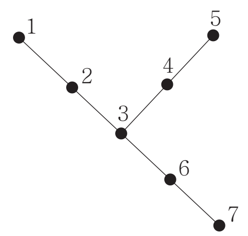

Figure 2: Graph having a perfect elimination order but is not an interval graph

5 Maximum clique size

The next question we will address is: How difficult is it to find the maximum clique number of a graph?

Recall the definition for maximum[1] clique number:

Definition 6 *The* ***maximum clique number*** *of a graph G, denoted $\omega(G)$, is the cardinality of the largest subset of V(G) that forms a clique.*

[1]Note the difference between the following two notations:
maximal clique : can't add edges to it and still get a clique,
maximum clique : the biggest possible of all cliques.

Theorem 4 *Computing $\omega(G)$ is NP-hard.*

The complete proof can be found in [GJ79], p.53-56. We will only give a short outline of the two steps of the proof:

1. Show that finding a clique is equivalent to finding a vertex cover: If I is a vertex cover of G then $V - I$ is a clique in $\bar{G}$.
2. Show that 3-SAT reduces to vertex cover.

We will now show that we can find $\omega(G)$ efficiently for interval graphs:

In the proof of Theorem 2 we have seen that $\text{Pred}(v_i^*) \cup v_i^*$ is a clique of size $\text{indeg}(v_i^*)+1$. So we have

$$\omega(G) \geq \text{indeg}(v_i^*) + 1$$

We know that for any graph

$$\chi(G) \geq \omega(G)$$

since it takes at least k colors to color a clique of size k. The proof for Theorem 2 showed that

$$\text{indeg}(v_i^*) + 1 = \chi(G).$$

So we have

$$\chi(G) \geq \omega(G) \geq \text{indeg}(v_i^*) + 1 = \chi(G).$$

which implies

Theorem 5 *For every graph G that has a perfect elimination order*

$$\chi(G) = \omega(G).$$

Therefore, one of the cliques we found is the maximum clique. So with the algorithm for coloring we also get the maximum clique size in $\mathrm{O}(m+n)$ time.

We also get:

Corollary 1 *For every induced subgraph of an interval graph*

$$\chi(G) = \omega(G).$$

Proof: Every induced subgraph of an interval graph G is an interval graph, since it corresponds to a subset of the set of intervals G was defined on.

□

References

[AH76] K. Appel and W. Haken. Every planar map is four colorable. *American Mathematical Society Bulletin*, 82(5):711–712, 1976.

[CH90] Fred C. Chow and John L. Hennessy. The priority-based coloring approach to register allocation. *ACM Transactions on Programming Languages and Systems*, 12(4):501–536, 1990.

[GJ79] M. R. Garey and D. S. Johnson. *Computers and Intractability - A guide to the theory of NP-completeness.* W H Freeman, 1979.

[Gol80] Martin Charles Golumbic. *Algorithmic graph theory and perfect graphs.* Academic Press, New York, 1980.

[Oga86] Hideo Ogawa. Labeled point pattern matching by delaunay triangulation and maximal cliques. *Pattern Recognition*, 19(1):35–40, 1986.

[Wer85] D. De Werra. An introduction to timetabling. european journal of operations research. *European Journal of Operations Research*, 19:151–162, 1985.