A GENERALIZATION OF CHORDAL GRAPHS

By

Jose M. Bonnin Cadogan

Thesis

Submitted to the Faculty of the

Graduate School of Vanderbilt University

in partial fulfillment of the requirements

for the degree of

Master of Science

in

Computer Science

August, 2011

Nashville, Tennessee

Approved:

Professor Jeremy Spinrad

Professor Mark Ellingham

TABLE OF CONTENTS

Introduction

A graph is a tuple $G = (V, E)$, where V is a set and E is a set of unordered pairs of distinct members of V [1]. In symbols, if $e \in E$, then $e = (v, w)$ for some $v, w \in V$ such that $v \neq w$. Many well-studied problems in graph theory are NP-complete. Researchers believe that an algorithm to solve one of these problems, or more precisely, an algorithm whose running time is polynomial as a function of the input, is likely to be very difficult if not impossible to find. There has been a great deal of research in developing algorithms to solve NP-complete problems in restricted classes of graphs, where finding a polynomial time solution could be more likely. This thesis introduces classes C and K, to be defined in sections three and four, and studies some NP-complete problems on these classes.

NP-Complete Problems

A clique C is a subset of V, the vertices of a graph $G = (V, E)$, such that if $v, w \in C$, then v and w are adjacent. An independent set I is a subset of V such that if $v, w \in I$, then v and w are not adjacent. *The maximum clique problem* consists of finding the maximum clique of a graph, and *the maximum independent set problem* is the problem of finding the maximum independent set of a graph. We denote the size of a maximum clique and the size of a maximum independent set of G, respectively, with $\omega(G)$ and $\alpha(G)$.

If each vertex v of a graph is associated with a weight $w(v)$, one can define the weight of an independent set as the summation of the weight of its vertices. The problem of finding an independent set with maximum weight is *the maximum weighted independent set problem.*

The clique cover problem consists of finding a minimum partition of the vertices of a graph into cliques. We denote the size of such partition with $\theta(G)$.

The coloring problem consists of finding a minimum partition of the vertices of a graph into independent sets. We denote the size of that partition with $\chi(G)$.

If $G = (V, E)$ is a graph, its complement is a graph $\overline{G} = (V, \overline{E})$, where $\overline{E} = \{(x, y) \mid x, y \in V \text{ and } (x, y) \notin E\}$. Solving the maximum clique problem in G is equivalent to solving the maximum independent set problem in $\overline{G}$, and solving the coloring problem in G is equivalent to solving the clique cover problem in $\overline{G}$.

Chordal graphs

Studying the coloring problem in general graphs might naturally lead to the idea of chordal graphs, a class of graphs with well-known results. This section introduces chordal graphs in that context. Moreover, all graph classes in this paper will be introduced after identifying a property that can help to solve some NP-complete problem.

Greedy Coloring

As defined in the introduction, the coloring problem consists of finding a minimum partition of the vertices of a graph into independent sets. Since partitions can be represented as functions, there is an equivalent formulation of the problem in terms of functions. A *coloring* c of a graph $G = (V, E)$ is a function from V to the set of positive integers. A coloring is *proper* if adjacent vertices of G are assigned different numbers (also referred to as colors). The coloring problem consists of finding a coloring with the minimum numbers of colors.

Remark If C is a clique of G, then $|C| \leq \chi(G)$.

The following greedy algorithm finds a proper coloring of any graph.

Color the vertices one by one so that the color of each vertex is the smallest positive integer not used by its colored neighbors.

Remark If we greedily color a vertex v, $c(v)$ uses a color in $[1, \delta(v) + 1]$.

Consider a vertex v whose neighborhood is a clique. $N[v]$ is a clique of size $\delta(v) + 1$, so $\delta(v) + 1 \leq \chi(G)$. It follows that if we greedily color v, $c(v) \leq \chi(G)$. We can use this fact to reduce the problem of coloring G to the problem of coloring $G - v$.

Remark If G is a subgraph of H, then $\chi(G) \leq \chi(H)$.

Remark Let c be a proper coloring of a graph G. If for every vertex v of G, $c(v) \leq \chi(G)$, then c is an optimum coloring.

Definition A simplicial vertex is a one whose neighborhood is a clique.

Let v be a simplicial vertex. We will show that the problem of coloring G can be reduced to the problem of coloring $G - v$. Let c be an optimum coloring of $G - v$. We color each vertex w different from v with color $c(w)$. We know that $c(w) \leq \chi(G - v) \leq \chi(G)$. Next, we greedily color v. Since $N[v]$ is a clique, $c(v) \leq \chi(G)$. The resulting coloring of G is optimum.

If we have a graph G in which the coloring problem can be successively reduced to the coloring problem in smaller graphs up to a trivial graph, then we can optimally color G by just greedily coloring simplicial vertices of induced subgraphs of G.

Definition Let G be a graph with n vertices. A *perfect elimination ordering* is a sequence of vertices $v_1, v_2, ..., v_n$ such that for every $i \in [1, n]$, v_i is a simplicial vertex in the subgraph of G induced by $v_i, v_{i+1}, ..., v_n$.

If a graph has a perfect elimination ordering, we can greedily color the vertices in the ordering from right to left and obtain an optimum coloring. Graphs with a perfect elimination ordering are called *chordal graphs*. The name comes from the fact that a graph has a perfect elimination ordering if and only if all its cycles of length four or greater have a chord [2]. A *chord* is an edge between non-consecutive vertices of a cycle.

The running time of the greedy coloring algorithm is $O(m+n)$, where m is the number of edges and n is the number of vertices of the graph being colored. Given a chordal graph, a perfect elimination ordering can be found in $O(m+n)$ time [3], so the coloring problem on chordal graphs can be solved in $O(m+n)$ time.

Maximum Clique Problem

The algorithm of the previous subsection used to optimally color a chordal graph can also be used to find its maximum clique. Consider the closed neighborhood $N[v]$ of a vertex v with color $\chi(G)$. $N[v]$ must have a size of at least $\chi(G)$, otherwise, the greedy algorithm would have chosen a smaller color-number for v. Moreover, since $\chi(G)$ is an upper bound to the size of any clique of G, $N[v]$ at the point when v is simplicial is a maximum clique.

Maximum Independent Set Problem

As observed before, if a graph has a simplicial vertex, one can reduce the coloring problem to the same problem in a smaller graph. This is also true for the maximum independent set problem.

Let v be a simplicial vertex of a graph $G = (V, E)$. Since $N[v]$ is a clique, an independent set of G can contain at most one vertex of $N[v]$. Consider a maximum independent set I of G. We can use $N[v]$ to partition I into two sets. We have $I = (I \cap N[v]) \cup (I \cap (V - N[v]))$. As noted,

$|I \cap N[v]| \leq 1$. Since $I \cap (V - N[v])$ is a subset of an independent set, it is also an independent set. Thus, $|I \cap (V - N[v])| \leq \alpha(G - N[v])$. We can now conclude $|I| \leq \alpha(G - N[v]) + 1$.

$\alpha(G - N[v]) + 1$ is not only an upper bound to I, but as shown next, also a lower bound. Consider a maximum independent set M of $G - N[v]$. Since M does not contain v or any of its neighbors, $M \cup \{v\}$ is an independent set of size $\alpha(G - N[v]) + 1$. The following theorem summarizes these results.

Theorem 1 *If v is a simplicial vertex of a graph G, then $\alpha(G) = \alpha(G - N[v]) + 1$. Morever, if M is a maximum independent set of $G - N[v]$, $M \cup \{v\}$ is an independent set of size $\alpha(G)$.*

Theorem 1 shows that the following greedy algorithm finds a maximum independent set of a chordal graph G.

Let I be an empty set. Consider the vertices of G one by one according to a perfect elimination ordering, from left to right. Let v be the vertex in consideration. If v has a neighbor in I, stop processing v. Otherwise, let v be a member of I.

The maximum independent set problem on chordal graphs can be solved in $O(m + n)$ time.

Clique Cover Problem

$\alpha(G)$ is a lower bound to $\theta(G)$. Thus, if a partition P of the vertices of G into cliques has size $\theta(G)$, then P is a minimum clique cover. Slightly modifying the greedy algorithm in the previous subsection, which finds a maximum independent set, allows us to find a minimum clique cover.

Let P be an empty partition. Consider the vertices of G one by one according to a perfect elimination ordering, from left to right. Let v be the vertex in consideration. If v is not a member of any block of P, let $N[v]$ be a block of P. When the algorithm terminates, P contains every vertex of G and has size $\alpha(G)$.

Class C

Theorem 1 proves that the maximum independent set problem on a graph G with a simplicial vertex v can be reduced to that problem on graph $G - N[v]$. The maximum independent set problem can be solved greedily on chordal graphs because all their induced subgraphs have a simplicial vertex. It is noteworthy, however, that it is not necessary for every induced subgraph to have a simplicial vertex. The algorithm still works if the subgraphs obtained by removing neighborhoods of simplicial vertices also have a simplicial vertex (or no vertices).

Definition A graph G is in *class* C if it is the null graph (the graph without vertices) or if G has a simplicial vertex v such that $G - N[v]$ is also in class C.

The maximum independent set problem and the clique cover problem can be solved in $O(m+n)$ time with the same greedy algorithms that solve these problems on chordal graphs, if the elimination scheme is given.

Recognition

A natural greedy algorithm to recognize if a graph is in class C successively removes neighborhoods of simplicial vertices of the graph until there are no vertices left. If at some point there is a non-null graph without any simplicial vertices, the algorithm terminates. It is possible to perform the removal process until there are no vertices if the input graph is in class C, but it is not obvious that the order in which the simplicial vertices are removed affects whether all vertices are removed when the algorithm ends. The next theorem proves that the order of removals is not important.

Theorem 2 *Graphs in class C can be recognized greedily*

Proof Let G be a graph in class C. Suppose the greedy algorithm does not remove all vertices. We represent the removal process with an ordering of the vertices of the graph. If a simplicial vertex of a subgraph of G is removed, the vertex precedes all other vertices of the subgraph, and each of its neighbors precedes all its non-neighbors. All vertices not removed by the algorithm are placed at the end of the ordering. Let σ_2 be the ordering of an execution of the algorithm that did not remove all vertices. Since G is in class C, there exists an ordering σ_1 in which all vertices of G are removed.

We call a vertex v of an ordering a *remover*, or an *actively removed*[1] vertex, if the algorithm removed its neighborhood because the vertex is simplicial. The vertices in $N(v)$ are *passively removed* vertices. If $w \in N(v)$, we say v *removed* w, and that v is a *remover* of w.

Let H be the subgraph of G induced by the vertices which were not removed in σ_2. Let y be the first vertex of σ_1 which is a member of H. By construction, the neighbors of y in H succeed y in σ_1.

$$\sigma_1 : v_1 v_2 ... y ... v_{n-1} ... v_n$$

$$\sigma_2 : v'_1 v'_2 ... \underbrace{y ... v'_{n-1} ... v'_n}_{\text{Vertices of H}}$$

If y is actively removed in σ_1, y is also actively removed in σ_2, because the neighbors of y that succeed it in σ_2 are a subset of the neighborhood removed from G by vertex y, and that neighborhood is a clique. Since y is not actively removed in σ_2 by definition, y is not actively removed in σ_1. Since every vertex in σ_1 is removed, y must have been removed by some vertex.

Let x be the remover of y in σ_1. x is not actively removed in σ_2, otherwise, it would have removed y, but y is not a removed vertex. Since x precedes y

[1]Terminology suggested by Mark Ellingham

in σ_1, $x \notin H$ (y is the first vertex of σ_1 which is in H). Thus, x precedes y in σ_2.

The expression below represents what we have proven so far. An asterisk above a vertex v, as in $\overset{*}{v}$, means that v is a actively removed. A minus sign above v, as in $\bar{v}$, means that v is passively removed.

$$\sigma_1 : v_1v_2...\overset{*}{x}...\bar{y}...v_{n-1}...v_n$$

$$\sigma_2 : v'_1v'_2...\bar{x}...\underbrace{y...v'_{n-1}...v'_n}_{\text{Vertices of H}}$$

The idea for the rest of the proof is to show that we can continue this process and prove the existence of an unbounded number of vertices.

Let r be the first vertex that is actively removed in σ_1 such that r removes a vertex s in σ_1 and r also precedes s in σ_2, where r is passively removed and s is not (i.e., s is either actively removed or not removed). x is an example of a vertex with that property.

$$\sigma_1 : v_1v_2...\overset{*}{r}...\bar{s}...v_{n-1}...v_n$$

$$\sigma_2 : v'_1v'_2...\bar{r}...s...v'_{n-1}...v'_n$$

Let q be the remover of r in σ_2. q is not adjacent to s, otherwise s would be a passively removed vertex in σ_2. Just suppose r precedes q in σ_1. It follows that q is adjacent to s, because r is actively removed in σ_1, and both

q and s are its neighbors. Since we showed that q is not adjacent to s, r does not precede q. If follows that q precedes r in σ_1, and since r is actively removed, q is not.

$$\sigma_1 : v_1 v_2 ... \overset{-}{q} ... \overset{*}{r} ... \overset{-}{s} ... v_{n-1} ... v_n$$

$$\sigma_2 : v'_1 v'_2 ... \overset{*}{q} ... \overset{-}{r} ... s ... v'_{n-1} ... v'_n$$

We can use a similar argument to the one used to show the existence of q to prove the existence of a vertex p that is actively removed in σ_1 and passively removed in σ_2. Let p be the remover of q in σ_1. p is not adjacent to r, otherwise r would be a passively removed vertex in σ_1. Just suppose q precedes p in σ_2. It follows that p is adjacent to r, because q is actively removed in σ_2, and both p and r are its neighbors. Since we showed that p is not adjacent to r, q does not precede p. If follows that p precedes q in σ_2, and since q is a remover, p is not.

$$\sigma_1 : v_1 v_2 ... \overset{*}{p} ... \overset{-}{q} ... \overset{*}{r} ... \overset{-}{s} ... v_{n-1} ... v_n$$

$$\sigma_2 : v'_1 v'_2 ... \overset{-}{p} ... \overset{*}{q} ... \overset{-}{r} ... s ... v'_{n-1} ... v'_n$$

p has the same property that defined r, but p precedes r. We have a contradiction which shows that ordering σ_2 does not exist.

Maximum Clique Problem is NP-complete

We will show that the maximum clique problem on general graphs can be reduced to the maximum clique problem on class C in polynomial-time. Then we will show that the maximum clique problem on class C is NP-complete.

Theorem 3 *The maximum clique problem on general graphs is polynomial-time reducible to the maximum clique problem on class C.*

Proof Let G be a graph with vertices $v_1, v_2, v_3, ..., v_n$. If $\omega(G) \leq 2$, a maximum clique can be found in polynomial time. Otherwise, we form a graph H which contains all the vertices and edges of G, but also contains a vertex v_i' for every vertex v_i. The only neighbor of a vertex v_i' in H is v_i. The following ordering certifies that H is a member of class C.

$$v_1', v_1, v_2', v_2, v_3', v_3, ..., v_n', v_n$$

We know that $\omega(H) \geq 3$, and no vertex of the form v_i' is in a maximum clique, because all such vertices are members of cliques that have size no greater than two. It follows that a maximum clique of H is also a maximum clique of G, and $\omega(H) = \omega(G)$.

Theorem 4 *The maximum clique problem on class C is NP-complete*

Proof We want to show that given a graph in class C, the problem of answering the following question Q (correctly) is NP-complete.

Does the graph have a clique of size k or greater?

First, we show the problem is NP. For every k which makes the correct answer 'yes', there exists a clique of size k, and one can verify that the clique is indeed one in polynomial-time. Next, we show that deciding Q is NP-hard.

It is NP-hard to decide whether a general graph has a clique of size k or greater [5], and we will show that this problem is polynomial-time reducible to deciding Q. To prove this, it suffices to show that the maximum clique problem on general graphs can be reduced in polynomial-time to the maximum clique problem on class C. This is enough because k has only n possible values. Since we have already shown the existence of such reduction, the maximum clique problem on class C is NP-complete.

Coloring Problem is NP-complete

We will show that the coloring problem on general graphs can be reduced to the coloring problem on class C in polynomial-time. Then we will show that the coloring problem on class C is NP-complete.

Theorem 5 *The coloring problem on general graphs is polynomial-time reducible to the coloring problem on class C.*

Proof Let G be a graph with vertices $v_1, v_2, v_3, ..., v_n$. If $\chi(G) < 2$, an optimum coloring can be found in polynomial time. Otherwise, we construct a graph H using the same procedure as in theorem 3.

Now we show $\chi(G) = \chi(H)$. Consider an optimum coloring of G. We color the vertices in H of the form v_i with the color of vertex v_i in G. We then greedily color every vertex of the form v_i'. Since every vertex of the form v_i' has color one or two, and every vertex of the form v_i has a color less than or equal to $\chi(G)$, H can be colored with $\chi(G)$ colors.

Theorem 6 *The coloring problem on class C is NP-complete*

Proof Given a graph in class C, we want to show that answering the following question Q is NP-complete.

Does the graph have a coloring using k or fewer colors?

The problem is in NP because a coloring is a valid certificate. Next, we need to prove that deciding Q is NP-hard, for which it suffices to show that the coloring problem on general graphs can be reduced in polynomial-time to the coloring problem on class C. Since we have already shown the existence of such reduction, the coloring problem on class C is NP-complete.

Class K

If a graph G is in class C, we can find a maximum independent set and an optimum coloring of G in polynomial time, as we have proven in previous sections. It follows that if $\overline{G}$ is in class C, the maximum clique problem and the clique cover problem can be solved in polynomial time.

Definition A graph G is in class K if G and $\overline{G}$ are members of class C.

Split graphs are those which are chordal and co-chordal (which means that their complements are chordal). Class K is a generalization of split graphs. All four problems mentioned in relation to class C can be solved on split graphs in polynomial time. A split graph has the property that its vertices can be partitioned into two sets such that one is a clique and the other one is an independent set (this property also defines split graphs) [4]. The property can be used to solve the maximum weighted independent set problem in polynomial time, so it is natural to ask whether that problem can be solved in class K. It turns out, however, that this problem is NP-complete.

Maximum Weighted Independent Set Problem is NP-complete

Theorem 7 *The maximum weighted independent set problem on class K is NP-complete.*

Given a graph in class K with nonnegative integer vertex weights, we want to show that answering the following question Q is NP-complete.

Does the graph have an independent set of weight w or greater?

The problem is in NP because an independent set of weight w or greater is a valid certificate.

Deciding whether a graph has an independent set of weight w is NP-hard. We will prove that this problem is polynomial-time reducible to deciding Q.

Given a graph G with weighted vertices, we construct a graph H which contains all vertices and edges of G. Moreover, for every vertex v_i, H contains vertices x_i and y_i. x_i is adjacent only to vertex v_i, while y_i is adjacent to every vertex of the form v_j except v_i, and every vertex of the form y_j. The construction of H is not complete, but we will consider an ordering of the vertices of H that have been defined.

$$x_1, x_2, x_3, ..., x_n, v_1, v_2, v_3, ..., v_n, y_1, y_2, y_3, ..., y_n$$

Successively eliminating neighborhoods of simplicial vertices according to the ordering above, from left to right, removes all the vertices. Note that for the purposes of this proof we do not require that the passively removed vertices immediately follow their removers.

x_1 is a simplicial vertex which removes v_1. Every vertex of the form x_i removes a vertex of the form v_i. Vertices $y_1, y_2, y_3, ..., y_n$ form a clique, so y_1 removes all of them. This ordering would show that the partial graph is in class C.

If we want to show that a graph is the complement of one in class C, instead of removing neighborhoods of simplicial vertices, we remove non-neighbors of vertices whose non-neighbors form an independent set. y_n cannot be removed, because x_n and v_n are among its non-neighbors, but x_n and v_n are adjacent. This suggests what we need at least one additional vertex so that the complement of H is in class C.

Let z be a vertex of H. z is adjacent to every vertex of the form v_i or y_i, but not adjacent to vertices of the form x_i. The construction of H is now complete.

The following ordering certifies that H is in class K. The only difference with the ordering above is that the new ordering contains z as its last vertex.

$$x_1, x_2, x_3, ..., x_n, v_1, v_2, v_3, ..., v_n, y_1, y_2, y_3, ..., y_n, z$$

Processing and removing vertices from left to right show that H is in class C, and processing and removing the vertices from right to left show that $\overline{H}$ is in class C.

Next, we set a weight for each vertex of H. Every vertex of H which is also in G has the same weight in H as in G. All other vertices of H have weight zero.

Let $\alpha_w(G)$ be the weight of a maximum weighted independent set J of H. We will show that $\alpha_w(H) \leq \alpha_w(G)$. Let I be an independent set G which contains all the vertices of J except those which are not in G. I and J have the same weight because all vertices of J which are not in I have weight zero, so if H contains an independent set of size w, so does G. Moreover, since G is a subgraph of H, H contains an independent set of weight w whenever G does, so deciding Q is NP-hard.

Open Problems

The greedy algorithm to recognize graphs in class C can run in $O(m \cdot n)$ time. Can a faster algorithm be designed?

How can graphs in class C or K be characterized? The construction in theorems 3 and 7 show that they do not have a characterization based on forbidden induced subgraphs, because every graph is an induced subgraph of a graph in class K (and therefore in class C).

The limit of the ratio between the number of split graphs and chordal graphs on n vertices is 1 [6]. What is the relationship between the number of graphs in class K and C?

References

[1] Jeremy Spinrad. *Efficient Graph Representations.* Fields Institute Monographs, American Mathematical Society, 2003.

[2] D. R. Fulkerson, O. A Gross. Incidence matrices and interval graphs, *Pacific J. Math 15*, pages 835-855, 1965.

[3] D. Rose, George Lueker, Robert E. Tarjan. Algorithmic aspects of vertex elimination on graphs, *SIAM Journal on Computing 5*, pages 266-283, 1976.

[4] Martin Charles Golumbic. *Algorithmic Graph Theory and Perfect Graphs*, Academic Press, Theorem 6.3, page 151, 1980.

[5] Michael R. Garey, David S. Johnson. *Computer and Intractability: A Guide to the Theory of NP-Completeness*, A Series of Books in the Mathematical Sciences, W. H. Freeman and Co, 1979.

[6] E. A. Bender, L.B. Richmond, N.C. Wormald, N. C., Almost all chordal graphs split, *J. Austral. Math. Soc., Series A 38 (2)*, pages 214-221, 1985.