Lecture 11

The Lovász ϑ Function*

11.1 Perfect graphs

We begin with some background on perfect graphs. First, we define some quantities on graphs.

Definition 11.1. Given a graph G on n vertices, we define the following quantities:

1. The *clique number* of G, written as $\omega(G)$, is the size of the largest clique in G.
2. The *independence number* of G, written as $\alpha(G)$, is the size of the largest independent set in G.
3. The *chromatic number* of G, written as $\chi(G)$, is the minimum number of colors required to properly color G.
4. The *clique cover number* of G, written as $\bar{\chi}(G)$, is the size of the smallest clique cover in G, which is the minimum number of vertex disjoint cliques such that every vertex is in some clique.

Recall that the complement of a graph G, denoted $\bar{G}$, is the graph on the same vertices as G such that two vertices are connected in $\bar{G}$ if and only if they are not connected in G.

The following facts will be useful:

1. $\alpha(G) = \omega(\bar{G})$
2. $\chi(G) = \bar{\chi}(\bar{G})$
3. $\omega(G) \leq \chi(G)$
4. $\alpha(G) \leq \bar{\chi}(G)$
5. $\alpha(G)\chi(G) \geq n$

*Lecturer: Anupam Gupta. Scribe: David Witmer.

The last fact holds because each color class is an independent set.

Now we give the definition of a perfect graph, first stated by Berge.

Definition 11.2. A graph G is *perfect* if $\omega(G') = \chi(G')$ for all vertex-induced subgraphs G' of G.

Example 11.3. Consider the 5-cycle C_5, shown in Figure 11.1. C_5 is its own complement, so have the following values for the quantities defined above:

$$\omega(C_5) = 2$$
$$\alpha(C_5) = 2$$
$$\chi(C_5) = 3$$
$$\bar{\chi}(C_5) = 3$$

C_5 is the smallest non-perfect graph.

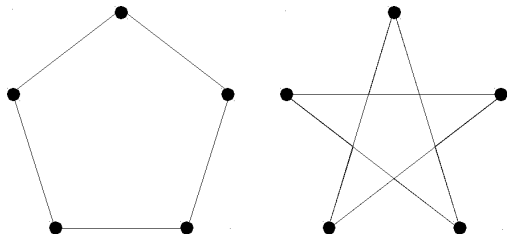

Figure 11.1: C_5 and $\bar{C_5}$. Note that C_5 is isomorphic to $\bar{C_5}$.

Bipartite graphs For any bipartite graph G, $\omega(G) = 2$ and $\chi(G) = 2$. Let $\mathrm{VC}(G)$ be the size of the minimum vertex cover of G. Then $\alpha(G) = n - \mathrm{VC}(G)$. By König's Theorem, this is equal to $n - \mathrm{MM}(G)$, where $\mathrm{MM}(G)$ is equal to the size of the maximum matching in G. In general, α is a lower bound for $\bar{\chi}$, but in this case, the two are equal. To see this, consider a clique cover of G consisting of 2-cliques corresponding to each edge of a maximum matching and 1-cliques for all remaining vertices as shown in Figure 11.2. The number of vertices not covered by the edges of the maximum matching is $n - 2\mathrm{MM}(G)$, so the number of cliques in this cover is $\mathrm{MM}(G) + (n - 2\mathrm{MM}(G)) = n - \mathrm{MM}(G)$. Then it must be true that $\bar{\chi}(G) \leq n - \mathrm{MM}(G)$, which, in turn, implies that $\alpha(G) = \bar{\chi}(G)$. This shows that both bipartite graphs and their complements are perfect.

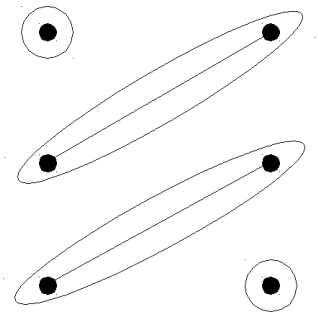

Figure 11.2: The clique cover corresponding to a matching of a bipartite graph.

Line graphs of bipartite graphs Recall that the line graph $L(G)$ of a graph G is the graph such that there is a vertex in $L(G)$ for each edge of G and two vertices of $L(G)$ are connected by an edge if and only their corresponding edges in G have a common endpoint. If G is bipartite, then $\omega(L(G))$ and $\chi(L(G))$ are both equal to the maximum degree of the vertices in G. In addition, $\alpha(L(G)) = \text{MM}(G)$ and $\bar{\chi}(L(G)) = \text{VC}(G)$. By König's Theorem, $\bar{\chi}(L(G)) = \text{MM}(G)$. Thus, line graphs of bipartite graphs and their complements are perfect.

Chordal graphs and interval graphs A chordal graph is a graph such that in every cycle of length four or more, there is an edge connecting two nonadjacent vertices of the cycle. Consider a set of intervals on the real line. The corresponding interval graph has a vertex for each interval and an edge between two vertices if the intersection of their intervals is nonempty. The set of interval graphs is a subset of the set of chordal graphs. An example of an interval graph is shown in Figure 11.3. Chordal graphs and their complements are perfect.

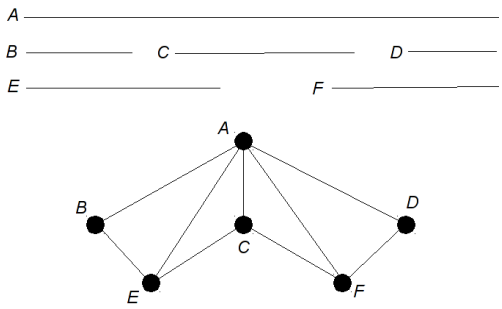

Figure 11.3: A set of intervals and the corresponding interval graph.

Comparability graphs Consider a partially ordered set P. The comparability graph of P is the graph with a vertex for each element of P and an edge between two elements if and only if their corresponding elements p and q in the partially ordered set are comparable ($p < q$ or $p > q$). Each clique in the comparability graph corresponds to a chain in the partially ordered set, and each independent set corresponds to an antichain. Let G be a

comparability graph. Then $\omega(G) = \chi(G)$. Consider the following coloring scheme: Choose a maximal antichain and color all of its elements one color. Remove these elements and continue inductively. Each time we remove a maximal antichain, the length of each maximal chain decreases by one, so $\omega(G)$ colors suffice. Since $\omega(G)$ is a lower bound for $\chi(G)$, we have equality. Also, $\alpha(G) = \bar{\chi}(G)$. Consider a maximal antichain. We can form a clique cover by taking the maximal chains containing the element of the antichain. Since $\alpha(G) \leq \bar{\chi}(G)$, the two quantities must be equal. Therefore, comparability graphs and their complements must be perfect.

For each of these classes of graphs, we see that their complements are also perfect. Lovász proved that this is true in general, a result known as the Weak Perfect Graph Theorem.

Theorem 11.4 (Weak Perfect Graph Theorem)**.** [**?**] *If G is a perfect graph, then its complement is also a perfect graph.*

Fulkerson had previously reduced the problem to showing that if G is perfect than G' is perfect, where G' is the graph formed by taking some vertex v, making a copy v' adjacent to all of the same vertices as v, and connecting v and v' by an edge. This is what Lovász proved.

Recall that C_5 is not a perfect graph. More generally, it is true that any odd cycle of length greater than or equal to five is not perfect. Such a cycle is called an odd hole. An odd antihole is the complement of one of these cycles. A Berge graph is a graph that contains no odd holes and no odd antiholes. The Strong Perfect Graph Theorem states that a graph is perfect if and only if it is a Berge graph.

Theorem 11.5 (Strong Perfect Graph Theorem)**.** [**?**] *A graph if and only if it is a Berge graph.*

Lovász gave an alternative characterization of perfect graphs:

Theorem 11.6. *A graph G is perfect if and only if for all induced subgraphs G', $\alpha(G')\omega(G') \geq n'$, where n' is the number of vertices in G'.*

Note that one direction is obvious: If G is perfect, then $\chi(G') = \omega(G')$, and it is always true that $\alpha(G')\chi(G') \geq n'$. Finally, it is also possible to check whether or not a graph is perfect in polynomial time.

Theorem 11.7. [**?**] *There exists a polynomial time algorithm to recognize perfect graphs.*

11.2 Computing α, ω, χ, and $\bar{\chi}$ for perfect graphs

Now we consider the problem of computing α, ω, χ, and $\bar{\chi}$ for perfect graphs. Assume we had a function $f(G)$ such that for all G, the following held:

$$\alpha(G) \leq f(G) \leq \bar{\chi}(G)$$

Then since $\alpha(G) = \bar{\chi}(G)$ for any perfect graph G, $f(G) = \alpha(G) = \bar{\chi}(G)$. If f were computable in polynomial time, we would be able to compute $\alpha(G)$ and $\bar{\chi}(G)$ for any perfect

graph G in polynomial time. We would be able to compute $\omega(G)$ and $\chi(G)$ by computing $f(\bar{G})$. We can make a first attempt at finding such an f using a linear program P.

$$\begin{aligned} \max & \sum_{i \in V} x_i \\ \text{s.t.} & \sum_{i \in C} x_i \leq 1 \quad \forall \text{ cliques } C \text{ in } G \\ & x_i \geq 0 \quad \forall i \in V \end{aligned} \tag{11.1}$$

Given a maximum independent set, setting $x_i = 1$ if i is in the set and $x_i = 0$ otherwise gives a feasible solution, so $\alpha(G) \leq \text{Opt}(P)$.

Consider the dual D:

$$\begin{aligned} \min & \sum_{\text{cliques } C} y_C \\ \text{s.t.} & \sum_{C \ni i} y_C \geq 1 \quad \forall i \in V \\ & y_C \geq 0 \quad \forall \text{ cliques } C \text{ in } G \end{aligned} \tag{11.2}$$

For a minimum clique cover, setting y_C to 1 if C is in the minimum clique cover ans $y_C = 0$ otherwise gives a feasible solution, so $\text{Opt}(D) \leq \bar{\chi}(G)$. This means that setting $f(G) := \text{Opt}(P) = \text{Opt}(D)$ satisfies $\alpha(G) \leq f(G) \leq \bar{\chi}(G)$ as desired.

However, we cannot solve these linear programs for general graphs. Consider the separation oracle that, given $x \in \mathbb{R}^{|V|}$ with $x \geq 0$ decides whether or not there exists some clique C such that $\sum_{i \in C} x_i \geq 1$. This solves the maximum weight clique problem, which is NP-hard. If we could solve P for general graphs, we would have such a separation oracle. This means that solving P must be NP-hard for general graphs. It is not clear now to solve D either, as it has an exponential number of variables.

Can we solve the P and D at least for perfect graphs? It is not clear even how to do that. So let's try using semidefinite programs.

11.3 The Lovász ϑ function

Lovász introduced a function ϑ satisfying $\alpha(G) \leq \vartheta(G) \leq \bar{\chi}(G)$ [**?**]. We begin by developing an SDP relaxation for $\bar{\chi}(G)$. We assign a unit vector v_i to each vertex. If two vertices are in the same clique of the minimum clique cover, we would like their vectors to be the same. If two vertices are not in the same clique, we would like their vectors to be as far apart as possible. Note that when k vectors are as spread out as possible, the dot product of any pair of them is $-\frac{1}{k-1}$. This means that if we have a clique cover of size k, there is an assignment of unit vectors to vertices such that every vertex in a clique is mapped to the same vector and, if two vertices are not in the same clique, the dot product of their vectors is $-\frac{1}{k=1}$. This is shown for clique covers of size 2, 3, and 4 in Figure 11.4.

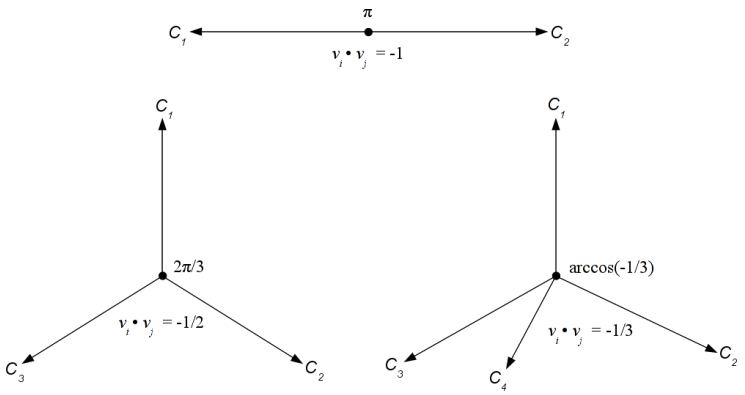

Figure 11.4: Assigning unit vectors to vertices such that the vertices in the same clique of the clique cover map to the same vector and vertices that are not in the same clique map to maximally separated vectors.

This suggests the following SDP relaxation:

$$\begin{aligned} &\min k \\ \text{s.t. } &\langle v_i, v_j \rangle = -\frac{1}{k-1} \quad i,j \in V, i \nsim j, i \neq j \\ &\langle v_i, v_i \rangle = 1 \qquad\qquad \forall i \in V \end{aligned} \tag{11.3}$$

where we use $i \sim j$ to denote that $(i,j) \in E(G)$, and $i \nsim j$ to denote that $(i,j) \notin E(G)$. We can now define the Lovász ϑ function.

Definition 11.8. Given $G = (V, E)$, $\vartheta(G)$ is the optimal value of the SDP in (11.3).

We can also write the following equivalent SDP:

$$\begin{aligned} &\min t \\ \text{s.t. } &\langle v_i, v_j \rangle = t \quad i,j \in V, i \nsim j, i \neq j \\ &\langle v_i, v_i \rangle = 1 \quad \forall i \in V \end{aligned}$$

In this case, the optimum is equal to $\frac{1}{1-\vartheta(G)}$. For a graph that is a clique, $-\frac{1}{k-1}$ and t both go to $-\infty$. Such graphs are not interesting to us, so this will not be a problem.

Theorem 11.9. $\alpha(G) \leq \vartheta(G) \leq \bar{\chi}(G)$

Proof. As described in the above discussion, any clique cover corresponds to a feasible solution of the SDP with an objective function value equal to the size of the clique cover. This implies that $\vartheta(G) \leq \bar{\chi}(G)$. It remains to show that $\alpha(G) \leq \vartheta(G)$. Suppose that $v_1, \ldots, v_s$ are the SDP solution vectors corresponding to a maximal independent set of size $s = \alpha(G)$ and let $v = \sum_{i=1}^{s} v_1$. Then $v^T v \geq 0$. It is also true that

$$v^T v = \left(\sum_{i=1}^{s} v_i\right)^T \left(\sum_{i=1}^{s} v_i\right) = \sum_{i=1}^{s} v_i^T v_i + \sum_{i \neq j} v_i^T v_j = s + \sum_{i \neq j} v_i^T v_j.$$

Then we have that $s + \sum_{i \neq j} v_i^T v_j \geq 0$. There are $s(s-1)$ terms in the sum, so, by averaging, there exist some distinct i and j such that

$$v_i^T v_j \geq -\frac{s}{s(s-1)} = -\frac{1}{s-1}.$$

Since $v_i^T v_j = -\frac{1}{\vartheta(G)-1}$ by the SDP constraints, $\alpha(G) = s \leq \vartheta(G)$. Therefore, we can conclude that $\alpha(G) \leq \vartheta(G) \leq \bar{\chi}(G)$. □

Note that $\vartheta(G)$ may not be rational for non-perfect graphs. For example, $\bar{\chi}(C_5) = 3$, $\alpha(C_5) = 2$, and $\vartheta(C_5) = \sqrt{5}$. So we cannot hope to get the exact optimum. However, we can solve the above semidefinite program to arbitrary accuracy using the ellipsoid algorithm, resulting in the following theorem.

Theorem 11.10. *For any $\epsilon > 0$, $\vartheta(G)$ can be computed to within ϵ error in time* $\text{poly}(n, \log \frac{1}{\epsilon})$.

The polynomial-time computability of the values of the parameters α, ω, χ, and $\bar{\chi}$ directly follows.

Corollary 11.11. *For any perfect graph G, $\alpha(G)$, $\omega(G)$, $\chi(G)$, and $\bar{\chi}(G)$ can be computed in polynomial time.*

11.3.1 Dual of the SDP

As an aside, the dual of the above SDP is the following SDP, with variables $B = (b_{ij})$:

$$\begin{aligned} \max \sum_{i,j \in V} & b_{ij} \\ \text{s.t. } & b_{ij} = 0 \quad i,j \in V, i \sim j \\ & \sum_{i \in V} b_{ii} = 1 \\ & B \succeq 0 \end{aligned} \tag{11.4}$$

We'll go through the process of deriving this dual program from the primal SDP in a future homework.

11.4 Non-perfect graphs and ϑ

We can also ask how closely ϑ approximated α for non-perfect graphs. Konyagin [**?**] constructed a graph G such that $\alpha(G) = 2$ and $\vartheta(G) = \Omega(n^{1/3})$, which is the largest that $\vartheta(G)$ can be. Alon and Kahale generalized this result with the following theorem.

Theorem 11.12. [**?**] *If $\alpha(G) \leq k$, then $\vartheta(G) \leq C n^{\frac{k-1}{k+1}}$ for some constant C.*

When α is not bounded, Feige showed the following result.

Theorem 11.13. [**?**] *There exists a graph G such that $\alpha(G) = n^{o(1)}$ and $\vartheta(G) = n^{1-o(1)}$.*

Håstad's results for the hardness of approximating the clique problem [**?**] also imply that such a graph must exist.

Kleinberg and Goemans showed that ϑ gives a 2-approximation for the size of the minimum vertex cover.

Theorem 11.14. [**?**] *For any graph G, $\frac{1}{2}\mathrm{VC}(G) \leq n - \vartheta(G) \leq \mathrm{VC}(G)$.*

This is not very useful for approximating $\mathrm{VC}(G)$, as the greedy algorithm for minimum vertex cover gives a 2-approximation. There are graphs for which this is tight, so we can do no better.

11.5 Finding cliques, independent sets, coloring, and clique covers of perfect graphs

Since we can compute α on perfect graphs, we can also find an independent set of size α in polynomial time. Consider the following algorithm: Remove a vertex from the graph. Calculate α for the resulting graph G'. If $\alpha(G') = \alpha(G) - 1$, put the removed vertex back; it must belong to the maximum independent set. Otherwise, leave it out. Repeat for the rest of the vertices in the new graph. The maximum independent set will remain at the end. We use the same method to find the maximum clique by noting that cliques are independent sets in the complement of the graph.

We next consider the problem if finding an optimal clique cover, which corresponds to an optimal coloring of the complement. In order to find an optimal clique cover, we need to find a maximum weight clique instead of a maximum size clique. To do this, we use a variant of the SDP in (11.4). Let w be non-negative vertex weights, which, without loss of generality, we assume to be integers. Then consider the following SDP:

$$\begin{aligned} \max \sum_{i,j \in V} \sqrt{w_i} b_{ij} \sqrt{w_j} & \\ \text{s.t. } b_{ij} &= 0 \quad i,j \in V, i \sim j \\ \sum_{i \in V} b_{ii} &= 1 \\ B &\succeq 0 \end{aligned} \tag{11.5}$$

We define $\vartheta(G, w)$ to be the optimal value of this SDP. Consider the graph G' formed by replacing each vertex i with a clique of size w_i such that two vertices in G' not in the same clique are adjacent if and only if the vertices in G corresponding to their cliques are adjacent. It is true that $\vartheta(G, w) = \vartheta(G')$. Let $\omega(G, w) = \omega(G'$ and $\chi(G, w) = \chi(G')$. Then $\omega(G, w) \leq \vartheta(\bar{G}, w) \leq \chi(G, w)$. Also, it is a fact that if G is perfect, then G' is perfect. In this case, $\omega(G, w) = \vartheta(\bar{G}, w) = \chi(G, w)$.. Therefore, by solving (11.5) and using self-reducibility as described above, we can find maximum weight cliques in perfect graphs.

We now give an algorithm to find a minimum clique cover, which corresponds to an optimal coloring of the complement.

Recall the primal-dual pair of linear programs P and D given above:

$$\begin{aligned} \max & \sum_{i \in V} x_i \\ \text{s.t.} & \sum_{i \in C} x_i \leq 1 \quad \forall \text{ cliques } C \text{ in } G \\ & x_i \geq 0 \quad \forall i \in V \end{aligned}$$

$$\begin{aligned} \min & \sum_{\text{cliques } C} y_C \\ \text{s.t.} & \sum_{C \ni i} y_C \geq 1 \quad \forall i \in V \\ & y_C \geq 0 \quad \forall \text{ cliques } C \text{ in } G \end{aligned}$$

These are the same as the linear programs (11.1) and (11.2) that we used in our initial attempt to find an f such that $\alpha(G) \leq f(G) \leq \bar{\chi}(G)$.

Step 1 Use the ellipsoid algorithm to solve the primal P. In order to do this, we need a separation oracle. Solving (11.5) to find the maximum weight of a clique gives us this separation oracle. The feasible region of P is a rational polyhedron, so we can find an optimal solution in polynomial time.

Step 2 Let $\mathcal{I} = \{C_1, C_2, \ldots, C_t\}$ be the set of polynomially-many cliques for which constraint were violated while running the ellipsoid algorithm in Step 1. Now consider the following linear program, which we will call $P_{\mathcal{I}}$:

$$\begin{aligned} \max & \sum_{i \in V} x_i \\ \text{s.t.} & \sum_{i \in C} x_i \leq 1 \quad \forall \text{ cliques } C \text{ in } \mathcal{I} \\ & x_i \geq 0 \quad \forall i \in V \end{aligned}$$

It is clear that $\text{Opt}(P_{\mathcal{I}}) \geq \text{Opt}(P)$. It cannot be true that $\text{Opt}(P_{\mathcal{I}}) > \text{Opt}(P)$. Otherwise, running the ellipsoid algorithm on the constraints in for the cliques in $\mathcal{I}$ would give $\text{Opt}(P) < \text{Opt}(P_{\mathcal{I}})$, which would contradict the correctness of the ellipsoid algorithm. So $\text{Opt}(P_{\mathcal{I}})$ must be equal to $\text{Opt}(P)$.

Step 3 Consider the dual of $P_{\mathcal{I}}$, which we will call $D_{\mathcal{I}}$. A feasible solution of $D_{\mathcal{I}}$ corresponds to a feasible solution of D in which all y_C such that $C \notin \mathcal{I}$ are set to 0. We know that $\text{Opt}(D_{\mathcal{I}}) = \text{Opt}(P_{\mathcal{I}})$, $\text{Opt}(P_{\mathcal{I}}) = \text{Opt}(P)$, and $\text{Opt}(P) = \text{Opt}(D)$, so $\text{Opt}(D_{\mathcal{I}}) = \text{Opt}(D)$. Now we can solve $D_{\mathcal{I}}$ in polynomial time to find an optimal solution of D. Call this solution y^*. Let C be some clique such that $y^*_C > 0$.

Step 4 By complementary slackness, if x^* is any optimal solution of P, then

$$\left(\sum_{i \in C} x_i^* - 1\right) y_C^* = 0.$$

Since $y_C^* > 0$, $\sum_{i \in C} x_i^* = 1$ for any optimal solution x^* of P. For any maximum independent set, let x be a solution to P such that $x_i = 1$ if i is in the set and $x_i = 0$ otherwise. For perfect graphs, $\alpha(G) = \text{Opt}(P) = \bar{\chi}(G)$, so x is an optimal solution to P. By the above, $\sum_{i \in C} x_i = 1$. This implies that all maximum independent sets of G contain exactly one vertex of C. Removing C from G therefore results in a graph G' such that $\alpha(G')$ is one less than $\alpha(G)$. Thus, recursing on G' gives us a clique cover of size $\bar{\chi}(G) = \alpha(G)$ as desired.

Bibliography

[AHK05] Sanjeev Arora, Elad Hazan, and Satyen Kale. The multiplicative weights update method: a meta algorithm and applications. Technical report, Princeton University, 2005.

[AK07] Sanjeev Arora and Satyen Kale. A combinatorial, primal-dual approach to semidefinite programs. In *STOC*, pages 227–236, 2007.

[GK07] Naveen Garg and Jochen Könemann. Faster and simpler algorithms for multicommodity flow and other fractional packing problems. *SIAM J. Comput.*, 37(2):630–652 (electronic), 2007.

[KL96] Philip Klein and Hsueh-I Lu. Efficient approximation algorithms for semidefinite programs arising from MAX CUT and COLORING. In *Proceedings of the Twenty-eighth Annual ACM Symposium on the Theory of Computing (Philadelphia, PA, 1996)*, pages 338–347, New York, 1996. ACM.

[Ste10] David Steurer. Fast sdp algorithms for constraint satisfaction problems. In *SODA*, pages 684–697, 2010.