Fast, Effective Vertex Cover Kernelization: A Tale of Two Algorithms[*]

Faisal N. Abu-Khzam[†], Michael A. Langston[‡] and W. Henry Suters[‡]

Summary

Two kernelization methods for the vertex cover problem are investigated. The first, *LP-kernelization*, has been in prior use and is known to produce predictable results. The second, *crown reduction*, is newer and faster but generates more variable results. Previously-unknown connections between these powerful methods are established. It is also shown that the problem of finding an induced crown-free subgraph in an arbitrary graph is decidable in polynomial time. Applications of crown structures are discussed.

1 Introduction

The difficulty and importance of finding solutions to $\mathcal{NP}$-hard problems has resulted in a wide variety of algorithmic techniques. One of the most common alternatives restricts attention only to polynomial-time approximation methods. In this paper we take a different approach, and exploit the fact that many of these problems are tractable when some key input parameter is fixed or

[*]This research has been supported in part by the National Science Foundation under grants EIA–9972889 and CCR–0075792, by the Office of Naval Research under grant N00014–01–1–0608, by the Department of Energy under contract DE–AC05–00OR22725, and by the Tennessee Center for Information Technology Research under award E01–0178–261.

[†]Division of Computer Science and Mathematics, Lebanese American University, Chouran, Beirut, Lebanon

[‡]Department of Computer Science, University of Tennessee, Knoxville, TN 37996–3450, USA

bounded, much as it often is in practice. The motivation for this general idea dates back to the Graph Minor Theorem and its many applications. See, for example, [3, 10, 13]. Since that time much has been learned about this strategy. Amenable problems are now termed "fixed parameter tractable" (henceforth FPT). For background and detailed information on FPT we refer the reader to [7].

If (S, k) is a problem instance in which S denotes some structure of size n and k is a parameter relevant to S, then the problem in question is said to be FPT if it can be solved by an algorithm whose run time is $O(f(k)n^c)$, where c is a constant independent of both n and k. Although these algorithmic bounds mean it is theoretically possible to find exact solutions in polynomial time as long as k is fixed, the associated constants of proportionality are often so excessive as to render FPT algorithms hopelessly infeasible from any practical standpoint. Thus a critical aspect in this approach relies on problem reduction, which in this setting is called "kernelization." As a general rule, the goal of kernelization is to take the aforementioned instance (S, k) and produce from it another instance (S', k') in which S' is of size $n' << n$ with some $k' \leq k$. This is of course done in such a way that (S', k') has a solution if and only if (S, k) has a solution.

Perhaps the best known example of an FPT problem, and the one we study here, is *vertex cover*. Given an undirected graph G and a parameter k, we seek to decide whether G contains a set C of k or fewer vertices such that every edge in G has at least one endpoint in C. Once kernelization has been accomplished, vertex cover can be solved in time $O(1.2852^{k'} + k'n)$ using a bounded search tree approach [5]. There are a myriad of known applications for this foundational combinatorial problem [2].

A variety of kernelization preprocessing rules have been proposed in an attempt to locate simple structures in an input graph. One rule is based on identifying vertices of high degree [4]. Others utilize vertices of low degree. With them we obtain a graph all of whose vertex degrees lie in the range [3,k'] and with $n' \leq \frac{k'^2}{3} + k'$. Put together these rules are straightforward to implement, but not especially effective because they can require quadratic time (in n) and produce kernels of quadratic size (in k). See [1] for details. We have no more to say about them here.

2 Techniques for Finding Linear-Sized Kernels

2.1 LP-Kernelization

One powerful, well-known kernelization technique is based on an integer programming formulation of the optimization version of vertex cover. We assign a weight $X_u \in \{0, 1\}$ to each vertex u of the graph $G = (V, E)$ so that the following conditions are met.

(1) Minimize $\sum_u X_u$.

(2) Satisfy $X_u + X_v \geq 1$ whenever $\{u, v\} \in E$.

We can relax this to a linear programming problem by replacing the constraint $X_u = \{0, 1\}$ with $X_u \geq 0$. The linear programming problem can be solved using a general LP package, or it can be posed as a network flow problem which can be solved using network flow techniques.

The solution to the linear programming problem is used to kernelize the original vertex cover problem in the following manner. Let $P = \{u \in V | X_u > 0.5\}$, $Q = \{u \in V | X_u = 0.5\}$ and $R = \{u \in V | X_u < 0.5\}$. There is an optimal vertex cover that is a superset of P and that is disjoint from R. This is a modification from [11] of a theorem originally proved in [12]. Furthermore, there is a solution to this problem in which $X_u = 0$ for all $u \in R$, $X_u = 1$ for all $u \in P$ and $X_u = 0.5$ for all $u \in Q$.

2.2 Crown Reduction

A much newer kernelization technique relies on the identification of a "crown structure" in the graph [1, 6]. The *crown reduction* method identifies two vertex subsets, H and I, in such a way that there is an optimal vertex cover with the property that it is both a superset of H and disjoint from I. Letting $N(S)$ denote the neighborhood of S, a *crown* is an ordered pair (I, H) of subsets of vertices from a graph G that satisfies the following criteria: (1) $I \neq \emptyset$ is an independent set of G, (2) $H = N(I)$, and (3) there exists a matching M on the edges connecting I and H such that all elements of H are matched. H is called the *head* of the crown. The *width* of the crown is $|H|$. See Figure 1.

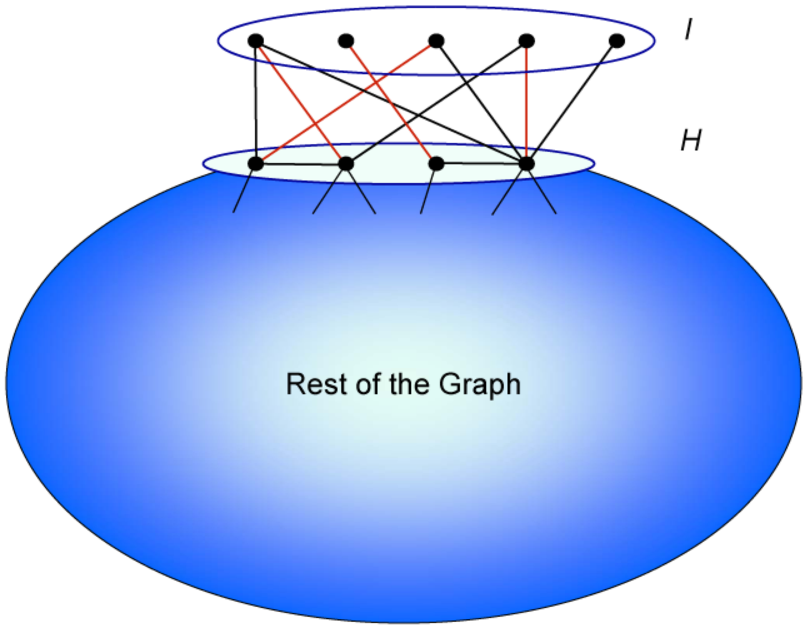

Figure 1: A sample crown decomposition.

A crown that is a subgraph of another crown is called a *subcrown*. A *straight crown* is a crown (I, H) that satisfies the condition $|I| = |H|$. A *flared crown* is a crown (I, H) that satisfies the condition $|I| > |H|$. These relationships are depicted in Figure 2. Notice that if (I, H) is a crown, then I is an independent set and H is a cutset between I and the rest of the graph.

The following algorithm can be used to find a crown in an arbitrary graph. It is listed here mainly for the reader's convenience. It is a refinement of the method originally proposed in [1], where a proof of its correctness and the conditions necessary to produce a nontrivial crown are given. Only Step 3 is new to this paper; its correctness relies merely on the definition of a crown. The previous algorithm was incapable of identifying straight crowns. The inclusion of Step 3 eliminates this restriction. It also allows the algorithm to find larger crowns than was previously possible. This algorithm will succeed in finding a crown as long as at least one of the following conditions is met: (1) I_0 is nonempty in Step 4 or (2) every vertex in $N(O)$ is matched in Step 3.

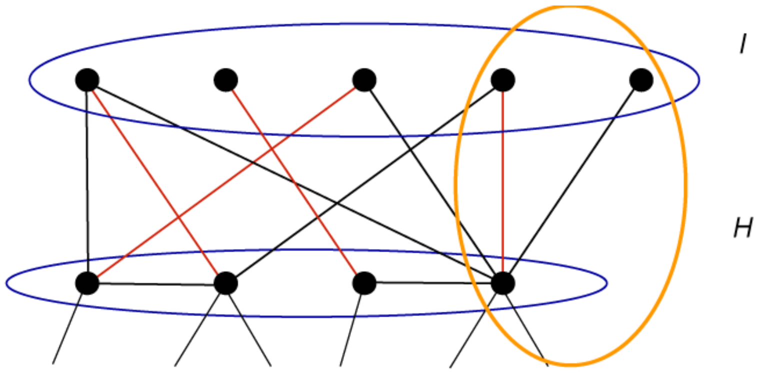

Figure 2: Selecting a subcrown to straighten a flared crown.

Let $N_M(H)$ denote the neighbors of H using only the edges contained in the matching M.

Step 1: Find a maximal matching M_1 of the graph, and identify the set of all unmatched vertices as the set O of outsiders.

Step 2: Find a maximum auxiliary matching M_2 of the edges between O and $N(O)$.

Step 3: If every vertex in $N(O)$ is matched by M_2, then $H = N(O)$ and $I = O$ form a crown, and we are done.

Step 4: Let I_0 be the set of vertices in O that are unmatched by M_2.

Step 5: Repeat steps 5a and 5b until $n = N$ so that $I_{N-1} = I_N$.

5a. Let $H_n = N(I_n)$.

5b. Let $I_{n+1} = I_n \cup N_{M_2}(H_n)$.

Step 6: $I = I_N$ and $H = H_N$ form a flared crown.

2.3 Performance of LP-Kernelization and Crown Reduction

The time complexity of LP-kernelization is $O(n^3)$ if a general LP package is used. When a network flow approach is used, it is is $O(m\sqrt{n})$ where m is the number of edges in G. The time complexity of crown reduction is $O(m^*\sqrt{n^*})$, where m^* is the number of edges between O and $N(O)$ and

n^* is the number of vertices in O and $N(O)$. Asymptotically, the behavior of the two methods is similar. In practice, however, m^* and n^* are generally much smaller than m and n. Extensive experimental results indicate that crown reduction is in fact much faster than LP-kernelization [1], especially on large problem instances.

Both LP-kernelization and crown reduction result in kernels whose sizes are linear in k. The kernel that results from LP-kernelization has size at most $2k$. The kernel that results from crown reduction is (perhaps loosely) bounded above by $3k$. It should be noted that the particular crown produced by crown reduction depends on the maximal matching identified in step 1 of the algorithm. Thus, the crown reduction may be repeated, potentially resulting in smaller and smaller kernels.

3 LP-kernelization: Finding Crowns

Because the crown reduction and LP-kernelization are based on different algorithmic techniques, there was considerable suspicion that the methods would prove to be orthogonal [9]. This is not the case, however. The sets P and R identified by LP-kernelization turn out, surprisingly, to be a crown. This is proven with the following theorem.

Theorem 1 *If LP-kernelization is applied to a graph G and finds a set R of vertices to exclude from the vertex cover and a set P of vertices to include in the cover, then (R, P) is a crown.*

Proof. Let us look at the requirements for a crown. Since $X_u < 0.5$ for all vertices $u \in R$, we know, because of the edge constraint $X_u + X_v \geq 1$, that there cannot be any edges $\{u, v\}$ with both u and v in R. Thus R is an independent set.

Suppose there is a vertex $u \in P$ where $u \notin N(R)$. Then every neighbor v of u has $X_v \geq 0.5$. Thus we could improve the LP solution by imposing $X_u = 0.5$ without violating any of the constraints $X_u + X_v \geq 1$. This cannot happen so we can conclude that $P \subseteq N(R)$. Similarly, suppose there is a $u \in N(R)$ where $u \notin P$. Then $X_u \leq 0.5$ while u has a neighbor v where $X_v < 0.5$. The edge $\{u, v\}$ must violate the constraint $X_u + X_v \geq 1$. This too cannot happen, so we conclude that $N(R) \subseteq P$ and thus $P = N(R)$.

Let M be a maximum matching on the edges between R and P. We now show that every vertex in P must be matched by contraction. Let $C_0 \subset P$ be the set of vertices in P that are unmatched by M and suppose $C_0 \neq \emptyset$. Let $D_0 = N(C_0) \cap R$ and $C_1 = N_M(D_0) \cup C_0$. Repeat this process, setting $D_n = N(C_n) \cap R$ and $C_{n+1} = N_M(D_n) \cup C_n$, until $C = C_{N+1} = C_N$ and $D = D_N$.

Since M is a maximum matching, alternating paths with an odd number of edges that begin and end at unmatched vertices are impossible. Thus any alternating path beginning with a vertex in C_0 has an even number of edges (and an odd number of vertices), beginning and ending in C. Since very vertex in D must be part of such an alternating path, this implies that C must be larger than D. This can be most easily seen by noting that the matching M gives a natural one to one association between the elements of D and $N_M(D)$. If this were not true an alternating path with an odd number of edges would result. Furthermore $C_0 \neq \emptyset$ and $N_M(D)$ are disjoint and $C = N_M(D) \cup C_0$. Thus C is larger than D.

Notice that for any set $P' \subset P$ we know that $N(P') \cap R$ must be larger than P' since otherwise we could improve the LP solution by setting $X_u = 0.5$ for all $u \in P'$ and $u \in N(P') \cap R$. However we have already shown that $C \subset P$ is larger than $N(C) \cap R = D$ so this is a contradiction. Thus $C_0 = \emptyset$ and every vertex in P must be matched. Therefore (R, P) is a crown. ■

3.1 Types of Crowns Identified

We now determine the types of crowns that can be identified by LP-kernelization. To do this, it is useful to prove the following lemma which helps refine the relationship between straight and flared crowns.

Lemma 1 *If (I, H) is a flared crown then there is another crown (I', H) that is straight and where $I' \subset I$.*

Proof. Since (I, H) is a crown there is a matching M on the edges between I and H so that all elements of H are matched. Since (I, H) is flared there must be at least one unmatched vertex in I. Let I' be the set of matched vertices in I. It is clear that $I' \subset I$ and that I' is an independent set. It is also clear that M is a matching between I' and H and that every vertex in H is matched. Thus (I', H) is a crown. Finally, the M forms a one-to-one association between the vertices in I' and H. Thus $|I'| = |H|$ and the crown is straight. ■

This lemma allows us to break a flared crown into two subcrowns. There is a straight subcrown and a flared subcrown. We are now ready to prove that LP-kernelization eliminates all flared crowns from the graph. It does this by either finding all of the flared crown, or by *straightening* the flared crown by identifying the flared part of the crown, but leaving the straight subcrown unrecognized. Since the LP-solutions are not unique, different solution techniques may produce different results on the same graph. One solution technique may identify an entire flared crown, while another technique may only straighten the crown.

Theorem 2 *If LP-kernelization is performed then the only straight crowns remain.*

Proof. Suppose there is a crown (I, H) with $|I| > |H|$ that is not identified by LP-kernelization. Vertices u not removed by LP-kernelization are given weights $X_u = 0.5$. Thus $X_u = 0.5$ for every $u \in I \cup H$. Since $|I| > |H|$ we can improved the LP-solution by assigning $X_u = 1$ for every $u \in H$ and $X_v = 0$ for every $v \in I$.

We must demonstrate that this new weight assignment is an LP-solution by showing that it still meets the edge constraints. Since $N(I) = H$ the condition $X_u + X_v \geq 1$ is satisfied by for all edges (u, v) where either u or v is in I. Next consider edges that have an endpoint in H but not in I. These edges have had their total weight increased and so still meet the edge constraints. Finally, edges that do not connect either to I or to H are unaffected by the new weights and so still satisfy the edge constraints. ■

3.2 Every Crown Identified by Some LP-Solution

Even though a given LP-solution may or may not recognize a particular straight crown, there is an LP-solution that does. This is proven in the following theorem.

Theorem 3 *If (I, H) is a crown, then there is an optimal LP-solution that identifies a crown of which (I, H) is a subcrown.*

Proof. Suppose there is a particular optimal LP-solution that does not identify a crown with (I, H) as a subcrown. This implies that the crown must be straight and $|I| = |H|$ by Theorem 2. Let us construct another LP-solution by assigning $X_v = 0$ for all $v \in I$ and $X_u = 1$ for all $u \in H$ and leaving the weight of the other vertices unchanged.

We first show that it is an LP-solution by showing that it still meets the edge constraints. Since $N(I) = H$ the condition $X_u + X_v \geq 1$ is satisfied by for all edges (u, v) where either u or v is in I. Next consider edges that have an endpoint in H but not in I. These edges have had their total weight increased and so still meet the edge constraints. Finally, edges that do not connect either to I or to H are unaffected by the new weights and so still satisfy the edge constraints.

Finally, we need to show that the solution is optimal. Since $|I| = |H|$ the value of the objective function $\sum_{u \in V} X_u$ is unchanged. Therefore this is a new solution to the LP-problem which identifies the crown.■

3.3 Finding All Crowns

Even though each crown has an LP-solution that would identify it, we need to determine if there is a particular LP-solution that would eliminate all crowns from the graph. We present a lemma and series of theorems that can be used to design a procedure for identifying all crowns in a graph.

First, we need to show that if a crown is identified and removed from a graph and a second crown is identified among the remaining vertices, then these two crowns can be combined to form a single crown. To do this it is useful to prove the following lemma that allows us to restrict the number of values that must be considered in the LP-solution. It is previously known that there are optimal solutions to the LP-problem that only use weights 0, 0.5, and 1 [11, 12]. Nevertheless, it is useful to recast this result here in term of crowns, using Theorem 1, and then to demonstrate a method for modifying any LP-solution to use only these weights.

Lemma 2 *If there is an optimal solution to the LP-kernelization problem that assigns weight X_u to each vertex $u \in V$ and we define $R = \{u \in V | X_u < 0.5\}$, $Q = \{u \in V | X_u = 0.5\}$, and*

$P = \{u \in V | X_u > 0.5\}$*, then there is another optimal solution to the LP-kernelization problem that assigns weights* $X'_u = 0$ *if* $u \in R$*,* $X'_u = 0.5$ *if* $u \in Q$*, and* $X'_u = 1$ *if* $u \in P$*.*

Proof. By Theorem 1 we know that (R, P) forms a crown and so there is a matching M between R and P so that every element in P is matched. The total weight contribution of $R \cup P$ must be at least $|M| = |P|$ in any LP-solution since the edges in M must have total edge weight ≥ 1. We can achieve this lower bound by making the weight assignments $X'_u = 0$ if $u \in R$ and $X'_u = 1$ if $u \in P$. The weights of the remaining vertices are unchanged by the assignment $X'_u = 0.5$ if $u \in Q$, so the total edge weight of the graph is either unchanged or reduced by these assignments.

We now show that this new assignment is in fact an LP-solution by considering the edge constraints. The edge constraints are met for all edges $\{u, v\}$ with $u \in P$, since in this case $X'_u = 1$. Edges $\{u, v\}$ with $u \in R$ must have $v \in P$ and so we have already met the edge constraint. This is because (R, P) is a crown and so $N(R) = P$. Finally, we consider edges $\{u, v\}$ with $u \in Q$. Such an edge cannot have $v \in R$ since $N(R) = P$ and so we only need to examine cases where $v \in P$ or $v \in Q$. If $v \in P$ then we have already met the edge constraint. If $v \in Q$ then $X'_u = 0.5$ and $X'_v = 0.5$ and so the edge constraint is met in this final case. ■

Now we prove that if a crown is identified and removed from a graph and a second crown is identified among the remaining vertices, then these two crowns can be combined to form a single crown.

Theorem 4 *Suppose a graph* $G = (V, E)$ *has a crown* (I, H) *identified by LP- kernelization and that when* (I, H) *is removed the induced subgraph* $G' = (V', E')$ *has another crown* (I', H')*, then* $(I \cup I', H \cup H')$ *forms a crown in* G*.*

Proof. Let S be the optimal LP-solution for G where $X_v = 0$ for every $v \in I$, $X_u = 1$ for every $u \in H$, and $X_w = 0.5$ for every $w \notin H \cup I$. We know that such an optimal LP-solution exists by Lemma 2.

We construct a new optimal LP-solution S'. For any $u \in V$ we set X'_u as follows:

(1) If $u \in H \cup H'$ then $X'_u = 1$.

(2) If $u \in I \cup I'$ then $X'_u = 0$.

(3) If $u \notin H \cup H' \cup I \cup I'$ then $X'_u = 0.5$.

Notice that all of the vertices in $I' \cup H'$ remain when (I, H) is removed so $I' \cup H'$ and $I \cup H$ are disjoint. We already know that I and H are disjoint and that I' and H' are disjoint. This implies that I, H, I', and H' are all mutually disjoint. Thus there are no contradictions in the definition of S'.

We now show that S' is in fact an LP-solution by verifying the edge constraints for an arbitrary edge $(u, v) \in E$.

Case 1: One of the endpoints is in $H \cup H'$. Without loss of generality assume $v \in H \cup H'$, then $X'_v = 1$ and the edge constraint is met.

Case 2: One of the endpoints is in I. Without loss of generality assume $u \in I$. We know that $v \in H$ since $N(I) = H$. Thus the problem reduces to case 1 and the edge constraint is met.

Case 3: One of the endpoints is in I'. Without loss of generality assume $u \in I'$. Since (I, H) is removed before (I', H') is identified, there are two possibilities $v \in I \cup H$ or $v \notin I \cup H$. If $v \in I \cup H$, we know that $N(I) = H$ and $u \notin H$ so we can restrict our attention to the case were $v \in H$. Thus the problem reduces to case 1 and the edge constraint is met. If $v \notin I \cup H$ then v is a vertex in G' and in this graph $N(I') = H'$. Thus $v \in H'$, the problem reduces to case 1 and the edge constraint is met.

Case 4: Neither u nor v is in $H \cup H' \cup I \cup I'$. In this case $X'_u = 0.5$ and $X'_v = 0.5$ and the edge constraint is met.

Finally, we show that S' is an optimal LP-solution for G. We know that S is an optimal LP-solution for G. Notice that the total edge weight in S is $|H| + 0.5|V - (H \cup I)| = |H| + 0.5|V'|$ and that an optimal LP-solution for G' has total edge weight $0.5|V'|$. Also notice that since (I', H') is a crown, there is another optimal LP-solution that identifies this crown and uses weights 0, 1, and 0.5. We know this is possible by the proof of Lemma 2. The total edge weight of this new LP-solution for G' is $|H'| + 0.5|V' - (H' \cup I')|$. However, since these are both optimal solutions for G' we know that $0.5|V'| = |H'| + 0.5|V' - (H' \cup I')|$.

Now consider the total edge weight for S' which is $|H \cup H'| + 0.5|V - (H \cup H' \cup I \cup$

$I')| = |H \cup H'| + 0.5|V - (H \cup I) - (H' \cup I')| = |H \cup H'| + 0.5|V' - (H' \cup I')|$. Since H, H' are disjoint, we know that $|H \cup H'| = |H| + |H'|$. Thus the total edge weight for S' is $|H| + |H'| + 0.5|V' - (H' \cup I')| = |H| + 0.5|V'|$ which is the total edge weight for S. Thus, both S and S' are optimal LP-solutions for G. Thus by Theorem 1 we know that the sets identified by S', namely $I \cup I'$ and $H \cup H'$ must form a crown. ■

Finally, we need to show that identifying two different crowns cannot result in any serious conflicts. That is, if a graph has two different crowns, then the two crowns can be combined to form a single crown. This is not a matter of simply taking the union of the two crowns. There may be vertices in the independent set of one crown that are included in the cutset of the other crown. Such conflicts always occur in straight crowns that are subcrowns of the two original crown and that can be reversed without disrupting the crown properties.

Theorem 5 *If (I_1, H_1) and (I_2, H_2) are crowns of a graph G that are identified by two different LP-solutions, then there is a crown (I, H) that contains all the vertices in $I_1 \cup I_2 \cup H_1 \cup H_2$ and where $I_1 \subseteq I$ and $H_1 \subseteq H$.*

Proof. For each vertex u in G, let X^1_u designate the weight of u in an optimal LP-solution that identifies (I_1, H_1) so that $X^1_u = 0$ if and only if $u \in I_1$, $X^1_u = 1$ if and only if $u \in H_1$, and $X^1_u = 0.5$ otherwise. We know such a solution exists by Lemma 2. Similarly find X^2_u for an optimal LP-solution that identifies (I_2, H_2).

We now create another optimal LP-solution by defining, for each vertex u, $X^*_u = \frac{X^1_u + X^2_u}{2}$. Notice the total weight $\sum_u X^*_u = \sum_u X^1_u = \sum_u X^2_u$ is still optimal. If $\{u, v\}$ is an edge in G, the $X^*_u + X^*_v = \frac{X^1_u + X^2_u}{2} + \frac{X^1_v + X^2_v}{2} \geq 1$ since $X^1_u + X^1_v \geq 1$ and $X^2_u + X^2_v \geq 1$. Thus X^* defines an optimal LP-solution. By Lemma 2 we can modify these values so that $X^*_u = 0$, 0.5, or 1 for all vertices u in G.

The only vertices in the original two crowns that do not have X^* weights of 1 or 0 are those in $I_1 \cap H_2$ and $I_2 \cap H_1$. Let G' be the graph that remains when all vertices with X^* weights of 0 or 1 are removed from the original graph. In this graph let $u \in I_1 \cap H_2$ and v be a neighbor of u. Since $u \in I_1$ and $H_1 = N(I_1)$ we know $v \in H_1$. Thus $X^1_v = 1$ and since we know $X^*_v = 0.5$ we also know $X_v^2 = 0$. Therefore $v \in I_2$ and $v \in H_1 \cap I_2$. Thus $N(I_1 \cap H_2) \subseteq (I_2 \cap H_1)$ in graph G'. A similar argument shows that $N(I_2 \cap H_1) \subseteq (I_1 \cap H_2)$ in G'.

Finally, we show that $(I_1 \cap H_2, I_2 \cap H_1)$ forms a straight crown that can be reversed. Notice that if $|I_1 \cap H_2| \leq |I_2 \cap H_1|$ we would not increase the size of the LP-solution by assigning weight 1 to the vertices in $I_1 \cap H_2$ and 0 to the vertices in $I_2 \cap H_1$. On the other hand if $|I_2 \cap H_1| \leq |I_1 \cap H_2|$ then we would not increase the size of the LP-solution by assigning weight 1 to the vertices in $I_2 \cap H_1$ and 0 to the vertices in$I_1 \cap H_2$. In either case we have a new LP-solution that identifies $(I_1 \cap H_2, I_2 \cap H_1)$ as a crown but X^* does not. By Theorem 2, this implies that $(I_1 \cap H_2, I_2 \cap H_1)$ is a straight crown. Since $N(I_1 \cap H_2) \subseteq (I_2 \cap H_1)$ and $N(I_2 \cap H_1) \subseteq (I_1 \cap H_2)$ in G' we know that this crown has no neighbors. Thus it can be reversed without loss of generality.

We select the independent set of the straight crown to be $I_1 \cap H_2$ and the cutset of the straight crown to be $I_2 \cap H_1$ so that the weight agree with the crown (I_1, H_1). Notice that all of the remaining vertices in the two original crowns were identified by the LP-solution defined by X^*. Thus, by Theorem 4, we can union the crown identified by X^* and the straight crown to obtain (I, H) where $I = I_1 \cup (I_2 - I_1)$ and $H = H_1 \cup (H_2 - I_1)$. ■

4 A Polynomial Time Algorithm to Identify All Crowns

We can now use the results we have just proven to produce a polynomial time algorithm that will find all possible crowns in an arbitrary graph.

Theorem 6 *There is a polynomial time algorithm for processing a graph G to produce an induced subgraph G' that has no crowns.*

Proof. We present such a polynomial time algorithm.

Step 1: Perform LP-kernelization. By Theorem 2, this can be used to eliminate all flared crowns by either removing the entire crown, or by removing enough vertices so that all the crowns are straight. Let $G_1 = (V_1, E_1)$ be graph induced by the set of vertices that remain in the kernel. Since these vertices were not removed by LP-kernelization, we know that $X_u = 0.5$ for all $u \in V_1$.

We also know that the total weight in the optimal LP-solution for G_1 is $0.5|V_1|$.

Step 2: Pick a vertex $w \in V_1$ and test it to see if it is in the independent set of some crown by finding the optimal solution of the following LP-problem.

Assign a value $X_u \geq 0$ to each vertex $u \in V_1$ so that the following conditions hold.

(1) Minimize $\sum_{u \in V_1} X_u$.

(2) Satisfy $X_u + X_v \geq 1$ whenever $uv \in E_1$.

(3) $X_w = 0$

Step 3: If the total weight is still $0.5|G_1|$, then this too is an optimal solution of the original LP-problem and we have identified a straight crown. We remove the straight crown from the graph in the usual manner producing an induced subgraph $G_2 = (V_2, E_2)$ where we know that $X_u = 0.5$ for all $u \in V_2$ and the total weight in the optimal LP-solution for G_2 is $0.5|V_2|$. If the total weight is larger, then we have not identified a crown and we let $G_2 = G_1$.

We repeatedly apply steps 2 and 3 until all vertices have been checked or eliminated, producing the graph G'. Theorem 4 guarantees that removing a crown does not create new crowns from vertices not previously in a crown. Theorem 5 guarantees that the order in which crowns are identified does not significantly change the final result. ■

Thus we only need to check each vertex once and when this process is complete, there can be no crowns and we will have identified all possible crowns in the graph. The total run time of the LP-solution procedure is $O(mn^{3/2})$ where m is the number of edges and n is the number of vertices in G if the network flow approach is used. This is a worst case scenario, in which the original graph has no crowns, and the process is repeated n times, once for each vertex.

5 Crown Decomposition

The algorithm in Theorem 6 allows us to find a single large crown that breaks the graph into a crown and a subgraph without any crowns. We state this in the form of the following corollary.

Corollary 1 *The union of the crowns found in the algorithm in Theorem 6 forms a single large crown and is, up to reversals of straight crowns, unique.*

Proof. Theorems 4 and 5. ■

This is equivalent to the following corollary that allows us to decompose every graph into a large crown and a subgraph that has no crowns.

Corollary 2 *Every graph G can be decomposed in to two subgraphs, C and K where C is a crown and K has no crowns.*

6 Concluding Remarks

Although the results reported here are primarily of a theoretical flavor, it would seem that in practice one would seek to kernelize a problem using one or more crown reductions before attempting the crown decomposition algorithm. Recent empirical investigations appear to bear out this presumption [1, 14]. Also, decomposing graphs into crowns and subgraphs that are crown free is not only useful in reducing the kernel size of the vertex cover problem. It may also be helpful in kernelizing a variety of packing problems, the $n - k$ coloring problem and many other $\mathcal{NP}$-hard problems [8]. Because the notion of crown decompositions is so new, there may well be other applications that have yet to surface.

References

[1] F. N. Abu-Khzam, R. L. Collins, M. R. Fellows, M. A. Langston, W. H. Suters, and C. T. Symons. Kernelization algorithms for the vertex cover problem: Theory and experiments. In *Proceedings, Workshop on Algorithm Engineering and Experiments (ALENEX)*, 2004.

[2] F. N. Abu-Khzam, M. A. Langston, P. Shanbhag, and C. T. Symons. Scalable parallel algorithms for FPT problems. Technical Report UT-CS-04-524, Department of Computer Science, University of Tennessee, 2004.

[3] D. Bienstock and M. A. Langston. Algorithmic implications of the graph minor theorem. In *Handbook of Operations Research and Management Science: Network Models*, pages 481–502. North-Holland, 1995.

[4] J. F. Buss and J. Goldsmith. Nondeterminism within $\mathcal{P}$. *SIAM Journal on Computing*, 22:560–572, 1993.

[5] J. Chen, I. Kanj, and W. Jia. Vertex cover: further observations and further improvements. *Journal of Algorithms*, 41:280–301, 2001.

[6] B. Chor, M. R. Fellows, and D. Juedes. Linear kernels in linear time, or how to save k colors in $O(n^2)$ steps. In *Proceedings, International Workshop on Graph Theoretic Concepts in Computer Science*, 2004.

[7] R. G. Downey and M. R. Fellows. *Parameterized Complexity*. Springer-Verlag, 1999.

[8] M. R. Fellows. personal correspondence.

[9] M. R. Fellows. Blow-ups, win/wins, and crown rules: Some new directions in FPT. *Lecture Notes in Computer Science*, 2880:1–12, 2003.

[10] M. R. Fellows and M. A. Langston. Nonconstructive tools for proving polynomial-time decidability. *Journal of the ACM*, 35:727–739, 1988.

[11] S. Khuller. The vertex cover problem. *ACM SIGACT News*, 33:31–33, June 2002.

[12] G.L. Nemhauser and L. E. Trotter. Vertex packings: Structural properties and algorithms. *Mathematical Programming*, 8:232–248, 1975.

[13] N. Robertson and P. D. Seymour. Graph minors XXIII, the disjoint paths problem. *Journal of Combinatorial Theory, Series B*, pages 65–110, 1995.

[14] W. H. Suters. Crown decomposition: Theoretical results and practical methods. Master's thesis, Department of Computer Science, University of Tennessee, 2004.