

Exploiting Planarity in Separation Routines for the Symmetric Traveling Salesman Problem

Adam N. Letchford¹ and Nicholas A. Pearson
Lancaster University

October 2006

Abstract

At present, the most successful approach to solving large-scale instances of the Symmetric Traveling Salesman Problem to optimality is *branch-and-cut*. The success of branch-and-cut is due in large part to the availability of effective *separation* procedures; that is, routines for identifying violated linear constraints.

For two particular classes of constraints, known as *comb* and *domino-parity* constraints, it has been shown that separation becomes easier when the underlying graph is planar. We continue this line of research by showing how to exploit planarity in the separation of three other classes of constraints: *subtour elimination*, *2-matching* and *simple domino-parity* constraints.

Keywords: traveling salesman problem, planar graphs, cutting planes.

1 Introduction

The well-known *Symmetric Traveling Salesman Problem* (STSP) is the problem of finding a minimum weight Hamiltonian circuit in an edge-weighted undirected graph. Although the STSP is strongly \mathcal{NP} -hard, many large-scale instances can be solved to optimality using the so-called *branch-and-cut* approach, see Padberg & Rinaldi [37], Applegate et al. [1] and Naddef [32]. This approach is based on the following integer programming formulation of the STSP, usually attributed to Dantzig, Fulkerson & Johnson [11]:

¹Corresponding author. Department of Management Science, Lancaster University Management School, Lancaster LA1 4YX, United Kingdom. E-mail A.N.Letchford@lancaster.ac.uk

$$\begin{aligned}
& \text{Minimise} && \sum_{e \in E} c_e x_e \\
& \text{Subject to:} && \\
& && x(\delta(\{i\})) = 2 \quad \forall i \in V, & (1) \\
& && x(\delta(S)) \geq 2 \quad \forall S \subset V : 2 \leq |S| \leq |V| - 2, & (2) \\
& && x \in \{0, 1\}^{|E|}. & (3)
\end{aligned}$$

Here, V is the vertex set, E is the edge set, c_e is the cost of traversing edge e , $\delta(S)$ denotes the set of edges with exactly one end-vertex in S and, as usual, $x(F)$ denotes $\sum_{e \in F} x_e$. It is common, but not necessary, to assume that the graph $G = (V, E)$ is complete. Equations (1) are called *degree equations*. The inequalities (2) are called *subtour elimination constraints* (SECs). The polyhedron defined by the degree equations, SECs and non-negativity constraints is sometimes known as the *subtour polytope*.

The key to the branch-and-cut approach is to use strong valid linear inequalities as cutting planes. These linear inequalities come from a study of the so-called *symmetric traveling salesman polytope*, which is the convex hull in $\mathbb{R}^{|E|}$ of vectors satisfying (1) - (3). Many classes of valid and even facet-inducing linear inequalities have been discovered. We refer the reader to the surveys Jünger, Reinelt & Rinaldi [25, 26] and Naddef [32] for a complete list. In this paper, we will refer only to the SECs themselves, and to *2-matching*, *comb*, *DP*, *simple comb*, *simple DP*, and *Chvátal comb* inequalities. Formal definitions and references are given in the next section. (See also Figure 2.)

To use inequalities in a given class as cutting planes, one needs a so-called *separation algorithm*. A separation algorithm is a procedure which, given a vector $x^* \in \mathbb{R}^{|E|}$ as input, either finds an inequality in the class which is violated by x^* , or proves that none exists (see Grötschel, Lovász & Schrijver [20]). Although many effective *heuristics* for separation are known for various classes of inequalities (see again the surveys [25, 26, 32]), the only *exact* separation results known are the following.

- The separation problem for the SECs is equivalent to a minimum weight cut problem, and can therefore be solved in $\mathcal{O}(nm + n^2 \log n)$ time using the algorithm of Nagamochi, Ono & Ibaraki [35]. (We use n and m to denote the number of vertices and the number of variables which are positive at x^* , respectively.)
- The separation problem for the 2-matching inequalities can be converted to a minimum weight *odd* cut problem on an expanded graph,

and therefore solved by the algorithm of Padberg and Rao [36]. A newer and faster algorithm, described by Letchford, Reinelt & Theis [30], runs in $\mathcal{O}(n^2 m \log(n^2/m))$ time.

- The separation problem for the simple DP inequalities can be reduced to a sequence of minimum weight odd cut problems, and thereby solved in $\mathcal{O}(n^3 m^3 \log n)$ time (Letchford & Lodi [28]). A faster implementation, running in $\mathcal{O}(n^2 m^2 \log(n^2/m))$ time, was recently given by Fleischer, Letchford & Lodi [16].
- Carr [5, 6, 7] showed that certain inequalities defined by *node-lifting* operations can also be separated in polynomial time, although the order of the polynomial is huge for most inequalities of interest.

However, more can be said in the interesting special case in which the fractional point defines a *planar graph*. More precisely, let E^* denote the set of edges whose variables are currently positive at x^* , i.e., $E^* = \{e \in E : x_e^* > 0\}$. (Note that $m = |E^*|$.) The *support graph*, usually denoted by G^* , is the subgraph of G induced by the edges in E^* , i.e., $G^* = (V, E^*)$. We say that x^* has *planar support*, or more briefly *is planar*, if the support graph G^* is planar.

Fleischer & Tardos [17] were the first to observe that planarity of x^* could be exploited. They gave an $\mathcal{O}(n^2 \log n)$ algorithm which, given a planar fractional point x^* , either finds a violated comb inequality or concludes that there are no comb inequalities violated by a *large amount* (we skip details for brevity). Inspired by that paper, Letchford [27] gave an $\mathcal{O}(n^3)$ exact separation algorithm for the DP inequalities, again for the case of planar support. These results are more useful than they might appear, because planar or near-planar fractional solutions often arise when solving real-life STSP instances (Boyd, Cockburn & Vella [4], Cook, Espinoza & Goycoolea [10]). Computational results given in [4, 10] show that the DP inequalities can be used to give extremely good lower bounds (typically within 0.1% of optimal) for large-scale STSP instances.

In this paper, we continue this line of research by describing fast separation algorithms for other inequalities in the planar case. In particular, we describe:

- an $\mathcal{O}(n \log^2 n)$ algorithm for the SECs;
- an $\mathcal{O}(n^{3/2} \log n)$ algorithm for the 2-matching inequalities;
- an $\mathcal{O}(n^2 \log n)$ algorithm for the simple DP inequalities.

These results, together with those in [17, 27], suggest that the STSP becomes somehow ‘amenable’ to solution via branch-and-cut (though still strongly \mathcal{NP} -hard) when the underlying graph is planar. This is in line with some other recent results in the literature, which suggest (from rather different viewpoints) that the planar STSP is somehow ‘relatively easy’:

- Arora et al. [2] gave a polynomial-time approximation scheme (PTAS) for the ‘planar metric’ STSP, in which the costs correspond to shortest paths in a weighted planar graph. (This includes the planar hamiltonian circuit problem as a special case.) Papadimitriou & Yannakakis [38] gave evidence that there is no PTAS for the general STSP.
- Deineko, Klinz & Woeginger [12] and Dorn et al. [14] gave dynamic programming algorithms for the ‘planar metric’ STSP which run in $\mathcal{O}(c^{\sqrt{n}})$ time, whereas the best known dynamic programming algorithm for the general STSP runs in $\mathcal{O}(n^2 2^n)$ time (Held & Karp [23]).

The structure of the remainder of the paper is as follows. In Section 2 we define the relevant valid inequalities in more detail. In Section 3 we explain the fast separation algorithms for SECs and 2-matching inequalities. In Section 4 we describe the algorithm for simple DP constraints, which is more complex. Some concluding remarks are given in Section 5.

2 Comb Inequalities and Variants

The most well-known constraints for the STSP, after the SECs themselves, are probably the *comb* inequalities of Grötschel & Padberg [21, 22]. These inequalities, which are facet-inducing for all $n \geq 6$, can be written in the form:

$$x(\delta(H)) + \sum_{j=1}^t x(\delta(T_j)) \geq 3t + 1, \quad (4)$$

where $t \geq 3$ is an odd integer and H and T_1, \dots, T_t are vertex sets satisfying:

$$\begin{aligned} T_i \cap T_j &= \emptyset \text{ for } 1 \leq i < j \leq t, \\ H \cap T_i &\neq \emptyset \text{ and } T_i \setminus H \neq \emptyset \text{ for } 1 \leq i \leq t. \end{aligned}$$

The set H is called the *handle* of the comb and the sets T_1, \dots, T_t are called the *teeth* (see Figure 1 for an illustration).

A number of special cases of the comb inequalities are of note. Comb inequalities satisfying $|H \cap T_i| = 1$ for all i had been previously discovered

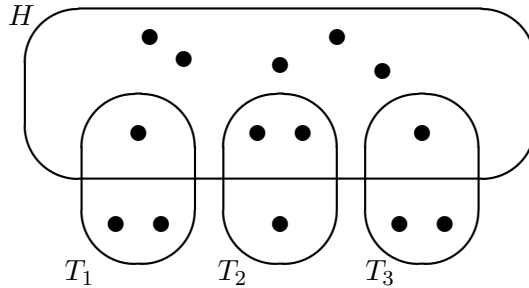


Figure 1: A comb with three teeth.

by Chvátal [9] and for that reason have come to be called *Chvátal comb inequalities*. The Chvátal comb inequalities themselves reduce to the *2-matching* (or *blossom*) inequalities of Edmonds [15] when $|T_i \setminus H| = 1$ for all i . (In this case, the teeth are mere edges.) Finally, Letchford & Lodi [28] call a comb *simple* if, for all i , either $|T_i \cap H| = 1$ or $|T_i \setminus H| = 1$ or both holds. The comb shown in Figure 1 is simple, but it is not a Chvátal comb, since $|T_2 \cap H| = 2$.

The comb inequalities in turn are a special case of the *domino-parity* (DP) inequalities, introduced by Letchford [27]. A *domino* is a pair $\{A, B\}$ of non-empty vertex sets such that $A \cap B = \emptyset$ and $A \cup B \neq V$. Let $t \geq 3$ be an odd integer as before. Given a handle H and dominoes $D_j = \{A_j, B_j\}$ for $j = 1, \dots, t$, the DP inequality takes the form:

$$x(F) + \sum_{j=1}^t x(\delta(A_j \cup B_j) \cup E(A_j : B_j)) \geq 3t + 1, \quad (5)$$

where the edge set F is defined in the following way: an edge e is in F if and only if exactly one of the following conditions holds:

- (i) it is in the cutset $\delta(H)$,
- (ii) $|\{j : e \in E(A_j : B_j)\}|$ is odd.

Comb inequalities are obtained when $A_j = T_j \cap H$ and $B_j = T_j \setminus H$ for all j . Although not every DP inequality induces a facet, there are many DP inequalities which induce facets yet are not comb inequalities [4, 27, 33, 34].

Finally, Fleischer, Letchford & Lodi [16] presented the *simple DP inequalities*. They say that a domino $\{A, B\}$ is simple if $|A| = 1$, or $|B| = 1$, or both. A simple DP inequality is a DP inequality in which all dominoes are simple.

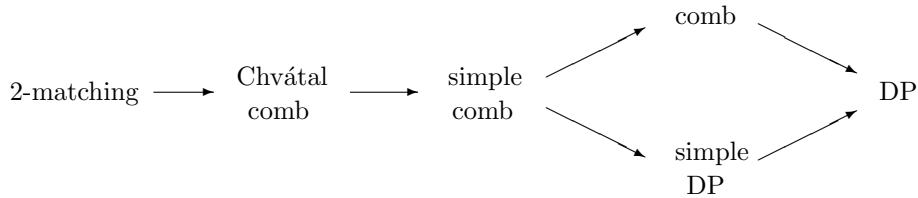


Figure 2: Relationships between various valid inequalities. An arrow from one class to another means that the latter is a proper generalization of the former.

To aid clarity, we show in Figure 2 the relationships between all of the inequalities discussed in this section.

3 Separation of SECs and 2-matching inequalities

In this section, we show how to exploit planarity in separation algorithms for the SECs and the 2-matching inequalities.

3.1 Subtour elimination constraints

From the definition of the SECs, it follows that an SEC is violated by a given x^* if and only if there is an edge cutset in the support graph G^* whose weight (computed with respect to x^*) is less than 2. Thus, any minimum weight cut algorithm can be used to solve the separation problem for the SECs. The minimum weight cut algorithm of Nagamochi, Ono & Ibaraki [35] runs in $\mathcal{O}(n(m + \log n))$ time. When G^* is sparse, i.e., when $m = \mathcal{O}(n)$, this reduces to $\mathcal{O}(n^2 \log n)$.

For planar graphs, however, faster minimum weight cut algorithms are known. Shih, Wu & Kuo [39] described an $\mathcal{O}(n^{3/2} \log n)$ algorithm and, very recently, Chalermsook et al. [8] found an $\mathcal{O}(n \log^2 n)$ algorithm. Both of these algorithms are based on the following two well-known ideas:

- Given any planar graph G , there exists another planar graph, the so-called (geometric or combinatorial) *dual* graph \bar{G} , with the following property: every edge cutset in G corresponds to a cycle in \bar{G} . Such a dual can be found in linear time. Thus, to find a minimum weight cut in G it suffices to find a minimum weight cycle in \bar{G} .

- In any planar graph $G = (V, E)$, one can find in linear time a vertex set $S \subset V$, called a *separator*, such that $|S| = \mathcal{O}(\sqrt{n})$ and such that the removal of S causes G to break into two disconnected components of approximately equal size (Lipton & Tarjan [31]). This leads naturally to a ‘divide-and-conquer’ approach to finding a minimum weight cycle in \tilde{G} .

3.2 2-matching inequalities

Although the separation algorithm of Letchford, Reinelt & Theis [30] is faster than that of Padberg & Rao [36], it turns out to be better to modify the Padberg-Rao algorithm in the planar case. The key to the Padberg-Rao algorithm is to write the 2-matching inequality in the form:

$$x(\delta(H) \setminus F) + \sum_{e \in F} (1 - x_e) \geq 1,$$

where H is the handle and $F \subset \delta(H)$ is the set of teeth. Then, the handle and the teeth define a violated inequality for x^* if and only if

$$x^*(\delta(H) \setminus F) + \sum_{e \in F} (1 - x_e^*) < 1.$$

Padberg and Rao then create a supergraph of G^* , the so-called *split graph*, by subdividing each edge into two ‘halves’. One half receives a weight equal to x_e^* and is labelled *even*, whereas the other half receives a weight equal to $1 - x_e^*$ and is labelled *odd*. Then, a violated 2-matching inequality exists if and only if there exists a cut in the split graph whose weight is less than 1, and which contains an odd number of odd edges. Padberg and Rao presented an algorithm to solve such minimum weight odd cut problems, which involves solving a sequence of $\mathcal{O}(m)$ max-flow problems. Using the well-known *pre-flow push* algorithm (Goldberg & Tarjan [18]) to solve the max-flow problems, along with some implementation tricks given in Grötschel & Holland [19], the Padberg - Rao separation algorithm can be implemented to run in $\mathcal{O}(nm^2 \log(n^2/m))$ time, which is $\mathcal{O}(n^3 \log n)$ in the planar case.

Clearly, the split graph is planar if and only if the original support graph is planar. Also, the split graph contains $\mathcal{O}(n)$ vertices and edges in the planar case. To compute the minimum weight odd cut in a planar graph, one can use the recent algorithm of the authors (Letchford & Pearson [29]), which runs in $\mathcal{O}(n^{3/2} \log n)$ time. Like the algorithms of [8, 39], this minimum weight odd cut algorithm uses planar duality to convert the problem into

a minimum weight odd cycle problem in the dual graph, and then uses the Lipton-Tarjan separator theorem to tackle this latter problem in a ‘divide-and-conquer’ manner.

If one is willing to separate 2-matching inequalities in $\mathcal{O}(n^2 \log n)$ time rather than $\mathcal{O}(n^{3/2} \log n)$ time, there is an alternative algorithm which avoids the computation of separators, and therefore is much simpler to implement. One simply uses the method of Barahona & Mahjoub [3] for finding a minimum weight odd cycle in the dual of the split graph. The algorithm of [3] involves calling the shortest-path algorithm of Dijkstra [13] $\mathcal{O}(n)$ times. The binary heap version of Dijkstra’s method, due to Williams [40], runs in $\mathcal{O}(n \log n)$ time on planar graphs and is very easy to implement.

4 Separation of Simple DP Inequalities

In this section we show that, when x^* is planar, the separation problem for simple DP inequalities can be solved in $\mathcal{O}(n^2 \log n)$ time. As in [27], we follow a ‘two-phase’ approach. In phase 1, we find a set of ‘candidate’ simple dominoes, i.e., simple dominoes whose ‘contribution’ to the slack of a simple DP inequality is sufficiently small to make it worthwhile considering them. In phase 2, we then test whether there is a violated simple DP inequality which uses some of the candidate simple dominoes.

As in [8, 27, 29, 39], we will be making heavy use of planar duality. If the embedding of G^* in the plane is fixed, there is a unique dual of G^* , which we will denote by \bar{G}^* .

We now describe each phase of the separation algorithm in turn.

4.1 Phase 1: finding candidate dominoes

Given a domino $\{A, B\}$, we define the edge sets:

$$\delta^*(A \cup B) = \delta(A \cup B) \cap E^*$$

and

$$E^*(A : B) = \{\{u, v\} \in E^* : u \in A, v \in B\}.$$

It is shown in [27] that a necessary condition for a domino $\{A, B\}$ to appear in a violated DP inequality is that the edge set $\delta^*(A \cup B) \cup E^*(A : B)$ forms three internally node-disjoint (s, t) -paths in \bar{G}^* (for suitable vertices s, t in \bar{G}^*). See Figure 3 for an illustration. Phase 1 of the planar DP separation algorithm involves computing, for each pair s, t in \bar{G}^* , a set of three disjoint (s, t) -paths of minimum total x^* -weight. This yields a so-called

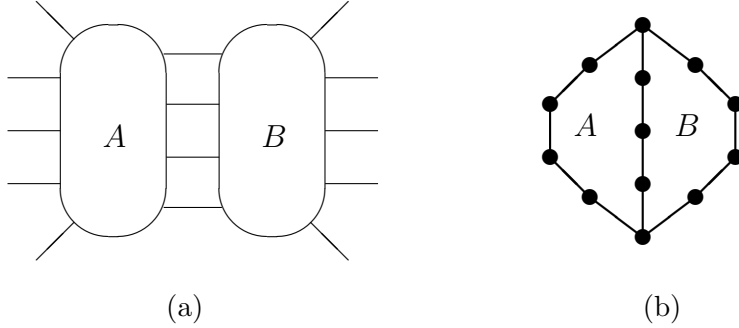


Figure 3: (a) edge set $\delta^*(A \cup B) \cup E^*(A : B)$ (b) three paths in dual.

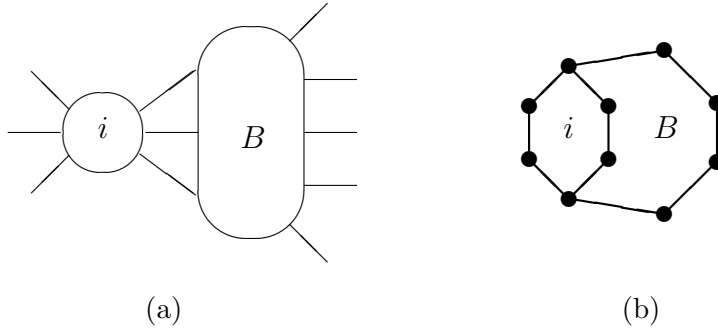


Figure 4: (a) edge set when $A = \{i\}$ (b) three paths in dual.

‘optimal’ domino for each pair s, t . It is shown in [27] that, if any violated DP inequality exists, then there exists a most-violated DP inequality which uses only optimal dominoes.

We will now show that the computation of the optimal dominoes can be performed in $\mathcal{O}(n^2)$ time when we impose the extra condition that the dominoes be simple. Let us assume without loss of generality that A is a singleton, say, $A = \{i\}$. The relevant edge set, $\delta^*(\{i\} \cup B) \cup E^*(\{i\} : B)$, is readily shown to be equal to

$$\delta^*(\{i\}) \cup E^*(B : V \setminus (B \cup \{i\})).$$

The edge-cutset $\delta^*(\{i\})$ corresponds to a face of \bar{G}^* , and the edge set $E^*(B : V \setminus (B \cup \{i\}))$ corresponds to a path in \bar{G}^* connecting two vertices of the face. See Figure 4 for an illustration.

Now, observe that, if x^* satisfies all degree equations, the term $x^*(\delta^*(\{i\}))$

is equal to 2. Thus, if we fix the vertex i and the dual vertices s and t , finding the optimal simple domino amounts to minimising the other term $x^*(E^*(B : V \setminus (B \cup \{i\})))$. This can be done by removing the edges in the face from \bar{G}^* , and then computing a shortest path from s to t in the remaining graph. Of course, if this is done in a naive way, an excessive number of shortest path computations will be needed. However, for a fixed i and s , it is possible to compute the optimal dominoes for all potential vertices t with a single call to a single-source shortest path algorithm. This immediately suggests the following algorithm for phase 1:

1. Assume that x^* lies in the subtour polytope. Construct a planar embedding of G^* and the corresponding dual graph \bar{G}^* .
2. For each vertex $i \in V$:
 - 2.1. Let $F(i)$ be the corresponding face of \bar{G}^* .
 - 2.2. Remove the edges of $F(i)$ from \bar{G}^* .
 - 2.3. For each vertex s lying in $F(i)$:
 - 2.3.1. Call a single-source shortest path algorithm with s as source.
 - 2.3.2. For each vertex $t \neq s$ lying in $F(i)$:
 - 2.3.2.1. Store s , t and the weight of the (s, t) -path.
 - 2.4. Add the edges of $F(i)$ back to \bar{G}^* .

This algorithm is illustrated in Figures 5 to 7. Figure 5(a) shows the support graph G^* for a fractional vector x^* for $n = 7$. Solid, dashed and dotted lines have weight 1, $2/3$ and $1/3$, respectively. The point is easily shown to violate a Chvátal comb inequality, and therefore a simple DP inequality. For clarity, G^* is redrawn in Figure 5(b), in which vertices are numbered from 1 to 7 and faces are labelled from ‘a’ to ‘f’. (Note that the outer face is labelled ‘f’). Figure 6(a) is a straight line embedding of the dual graph \bar{G}^* , where the letters now represent vertices.

Suppose we select $i = 3$. The corresponding face of \bar{G}^* is bounded by vertices a, b, c and d and is labelled $F(3)$ in Figure 6(a). Figure 6(b) depicts the subgraph of \bar{G}^* obtained by removing the edges of $F(3)$. Now suppose that we choose vertex a as our source vertex. The shortest path tree with source a is shown in figure 7(a). This immediately yields the three shortest paths a-f-b, a-e-c, a-e-d, which correspond to three optimal simple dominoes. The path a-e-c, for example, when added to the face $F(3)$, yields the configuration displayed in Figure 7(b). This corresponds to an optimal simple domino in which $i = 3$ and B is the set of vertices bounded in G^* by the faces a, d, c and e, i.e., $B = \{4, 5\}$.

We now analyse the running time of this version of phase 1.

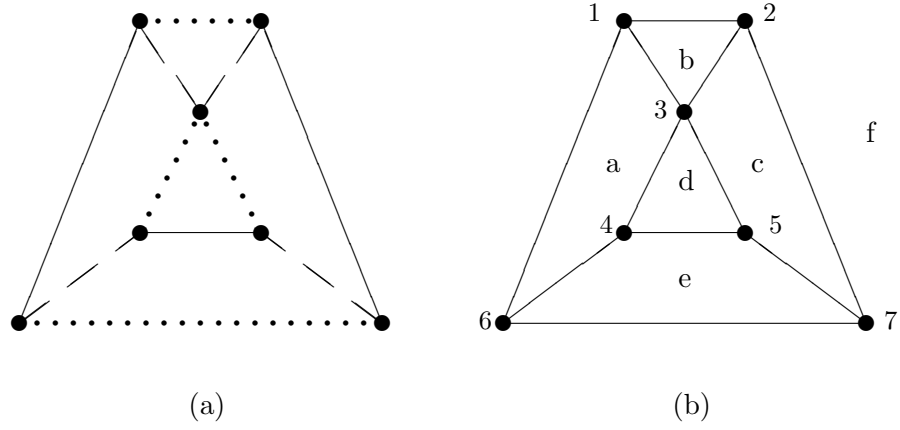


Figure 5: (a) Support graph of a fractional point and (b) a labelling of its vertices and faces.

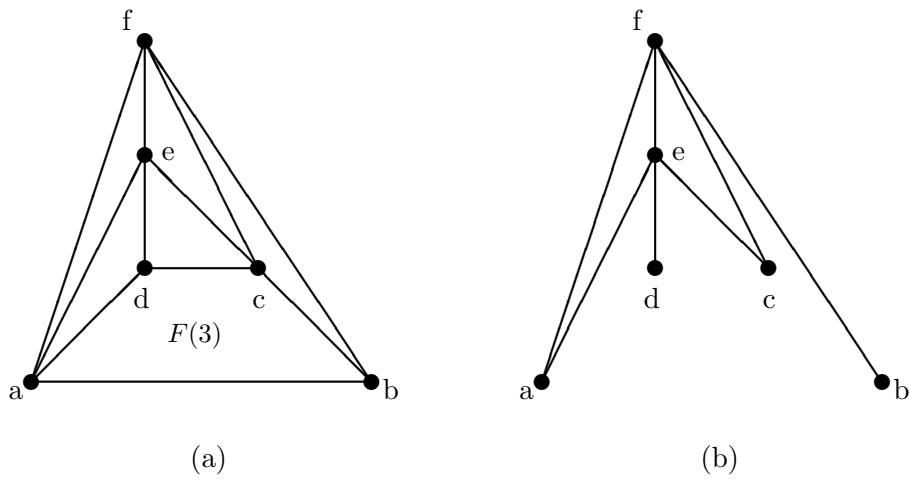


Figure 6: (a) dual graph \bar{G}^* and (b) \bar{G}^* with edges of $F(3)$ removed.

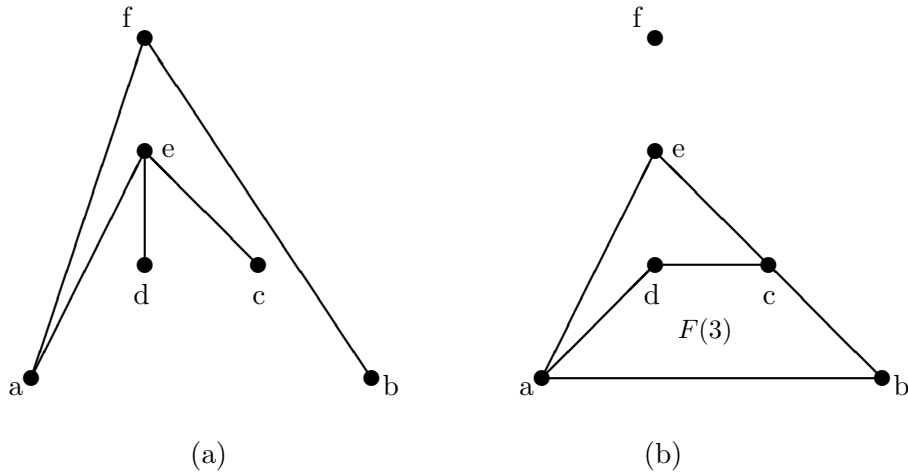


Figure 7: (a) Shortest path tree rooted at a .
(b) Induced simple domino, with $A = \{3\}$ and $B = \{4, 5\}$.

Lemma 1 *Phase 1 of simple DP separation can be performed in $\mathcal{O}(n^2)$ time in the planar case.*

Proof. A planar embedding and the associated dual can be found in linear time. For a fixed i and a fixed vertex s in the face, we can call the linear-time planar single-source shortest-path algorithm of Henzinger et al. [24]. For a fixed i , the number of such calls is equal to the number of edges in the corresponding face of \bar{G}^* . Thus, the total number of shortest-path calls is equal to $2|E^*|$. Since G^* is planar, $|E^*|$ is $\mathcal{O}(n)$. \square

In practice, we would recommend using the binary heap version of Dijkstra's method to compute the shortest paths in phase 1. As mentioned above, it runs in $\mathcal{O}(n \log n)$ time on planar graphs, and it is far easier to implement than the algorithm of [24]. The resulting version of phase 1 runs in $\mathcal{O}(n^2 \log n)$ time. This increase of a $\log n$ factor in phase 1 has no effect on the overall running time bound, since phase 1 is not the bottleneck of the algorithm.

4.2 Phase 2: finding the best handle

Now we briefly review phase 2 of the algorithm of [27]. The 'weight' of an optimal domino is defined as the sum of the weights of the three (s, t) -paths, minus 3. It is shown in [27] that, if x^* lies in the subtour polytope, then the weight of all dominoes is non-negative, and that any domino with a weight

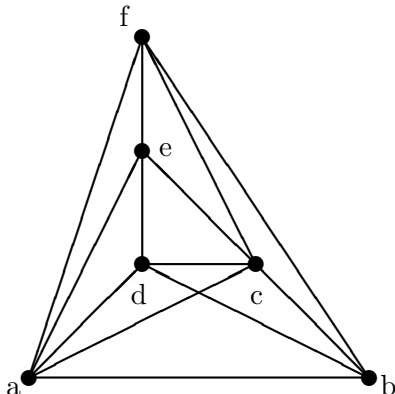


Figure 8: Supergraph \bar{G}^+ for previous example (parallel edges omitted).

of 1 or more can be discarded. A labelled supergraph of \bar{G}^* , which we will denote by \bar{G}^+ , is then constructed as follows. For each remaining optimal domino, an additional edge is added to \bar{G}^* connecting the corresponding vertex pair (s, t) . The edge is labelled ‘odd’ and given a weight equal to the weight of the associated optimal domino. A violated DP inequality then exists if and only if there exists an odd cycle (i.e., a cycle containing an odd number of odd edges) of weight less than 1 in \bar{G}^+ .

When we restrict attention to simple dominoes, things simplify a little. First, when all degree equations are satisfied, the weight of the optimal simple domino for a given pair (s, t) is equal to the weight of the single (s, t) -path found in phase 1, minus 1. Moreover, an odd edge can only exist in \bar{G}^+ if its end-vertices lie on the same face of \bar{G}^* . Unfortunately, \bar{G}^+ can still be non-planar, as shown in Figure 8.

The key to obtaining a fast algorithm for phase 2 is to borrow a concept from [28]. They say that an optimal simple domino is *light* if its weight is less than $1/2$, and *heavy* if its weight is at least $1/2$ but less than 1. In our context, a domino is light if the associated (s, t) -path has a weight less than $3/2$, and heavy if it has a weight greater than or equal to $3/2$ and less than 2. We will say that an (s, t) -path itself is light or heavy accordingly. We denote by $\bar{G}^{1/2}$ the supergraph of \bar{G}^* obtained by adding odd edges only for the light simple dominoes. (Clearly, $\bar{G}^{1/2}$ is a subgraph of \bar{G}^+ .) We have the following theorem:

Theorem 1 $\bar{G}^{1/2}$ is planar.

Proof. In [28] it was shown that it is impossible for there to exist two light dominoes $\{\{i\}, B\}$, $\{\{i\}, B'\}$ which *cross*, i.e., such that all of the vertex sets

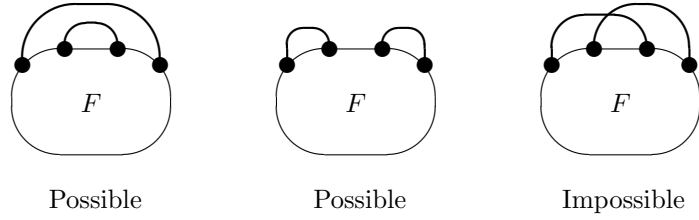


Figure 9: For a fixed face F of \bar{G}^* , two light (s, t) -paths cannot cross.

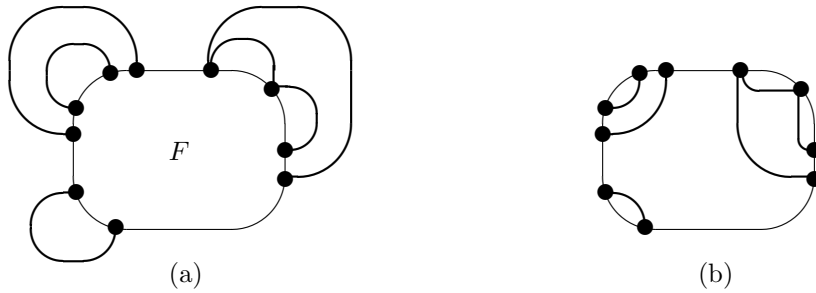


Figure 10: (a) Light (s, t) -paths associated to a fixed face F .
(b) Corresponding odd edges embedded in the plane inside F .

$B \cap B'$, $B \setminus B'$, $B' \setminus B$ and $V \setminus (B \cup B' \cup \{i\})$ are non-empty. This means that, for a given face F in step 2 of our algorithm for phase 1, it is impossible for two light (s, t) -paths to share an internal vertex in common. (See Figure 9 for an illustration.) Thus, for a given face F , the extra odd edges added to \bar{G}^* to represent light (s, t) -paths can be embedded in the plane inside F without crossing (Figure 10.) Doing this for each face conserves planarity. Hence $\bar{G}^{1/2}$ can be embedded in the plane. \square

Although it is not crucial to the overall argument, the following corollary is of independent interest:

Corollary 1 *If x^* is planar and phase 1 has already been performed, we can detect if a violated simple DP inequality exists which uses only light simple dominoes in $\mathcal{O}(n^{3/2} \log n)$ time.*

Proof. $\bar{G}^{1/2}$ has $\mathcal{O}(n)$ vertices and is planar. To test if such a violated inequality exists, it suffices to find a minimum weight odd circuit in $\bar{G}^{1/2}$. This can be done in $\mathcal{O}(n^{3/2} \log n)$ time by the algorithm in [29]. \square

To deal with the heavy simple dominoes, we use two more results proved in [28]. First, a violated DP inequality can have at most one heavy domino. Second, it is never necessary to use two simple dominoes of the form $\{\{i\}, B\}$, $\{\{i\}, B'\}$ with the same ‘root’ vertex i . This implies that one should never use two odd edges lying in the same face of \bar{G}^* . This leads to the following algorithm for phase 2.

1. Assume that G^* , \bar{G}^* and the optimal simple dominoes are already available from phase 1.
2. For each light simple domino, add an odd edge to \bar{G}^* , leading to the labelled weighted supergraph $\bar{G}^{1/2}$.
3. For each face f of \bar{G}^* :
 - 3.1. Remove the odd edges connecting vertices in that face from $\bar{G}^{1/2}$.
 - 3.2. For each vertex s lying in the face:
 - 3.2.1. Add odd edges to $\bar{G}^{1/2}$ connecting s to other vertices in the face (regardless of whether the associated domino is light or heavy).
 - 3.2.2. Find a minimum weight odd circuit passing through s in $\bar{G}^{1/2}$.
 - 3.2.3. If the odd circuit has a weight less than 1, output the violated simple DP inequality.
 - 3.2.4. Remove the odd edges which were added in step 3.2.1.
 - 3.3. Add back the odd edges which were removed in step 3.1.

The analysis of the running time for this version of phase 2 is fairly straightforward.

Lemma 2 *Phase 2 of simple DP separation can be performed in $\mathcal{O}(n^2 \log n)$ time in the planar case.*

Proof. As in the proof of Lemma 1, step 3.2 is performed $\mathcal{O}(n)$ times. Each time step 3.2 is called, the bottleneck is the minimum weight odd circuit problem in step 3.2.2. As noted in [29], such a minimum weight odd circuit can be computed in $\mathcal{O}(n \log n)$ time with one Dijkstra call, using the method of Barahona & Mahjoub [3]. \square

Thus, we have proved:

Theorem 2 *When G^* is planar and lies in the subtour polytope, the separation problem for simple DP inequalities can be performed in $\mathcal{O}(n^2 \log n)$ time.*

Proof. By Lemma 1, phase 1 can be performed in $\mathcal{O}(n^2)$ time. By Lemma 2, phase 2 can be performed in $\mathcal{O}(n^2 \log n)$ time. Clearly, phase 2 is the bottleneck of the overall algorithm. \square

As well as being very fast, the new algorithm is relatively easy to implement. The core subroutine needed is merely a binary heap version of Dijkstra’s method.

We close this section with the following conjecture:

Conjecture 1 *If x^* is planar, a simple DP inequality is violated if and only if a simple comb inequality is violated.*

Note that a similar result does *not* hold for non-planar points x^* ; see [16] for a counter-example.

5 Conclusions

The goal of this paper has been to build on the results of [17, 27], showing that the separation problem for various valid inequalities becomes a lot easier if the fractional point to be separated has planar support. Table 1 summarizes the results discussed. The column headed ‘general’ gives the worst-case running time for general graphs. The column headed ‘sparse’ shows how these times simplify for sparse graphs (i.e., graphs for which $m = \mathcal{O}(n)$). Finally, the column headed ‘planar’ gives the corresponding times for planar graphs. It is obvious that significant gains can be made by exploiting planarity.

Inequalities	General	Sparse	Planar
Subtour elimination	$\mathcal{O}(nm + n^2 \log n)$	$\mathcal{O}(n^2 \log n)$	$\mathcal{O}(n \log^2 n)$
2-matching	$\mathcal{O}(n^2 m \log(n^2/m))$	$\mathcal{O}(n^3 \log n)$	$\mathcal{O}(n^{3/2} \log n)$
Simple DP	$\mathcal{O}(n^2 m^2 \log(n^2/m))$	$\mathcal{O}(n^4 \log n)$	$\mathcal{O}(n^2 \log n)$
DP	unknown	unknown	$\mathcal{O}(n^3)$

Table 1: Separation of inequalities for planar STSP: new status.

There are some interesting open questions. Among them, the most pressing seems to be the following: can the separation problem for general (i.e., non-simple) DP inequalities be solved in polynomial time, on general support graphs? A less ambitious goal would be to find a DP separation algorithm for large and interesting superclasses of planar graphs, such as graphs with no $K_{3,3}$ minor or graphs with no K_5 minor.

Acknowledgement: The authors are grateful to the anonymous referees and to Dirk Oliver Theis for detailed lists of corrections.

References

- [1] D. Applegate, R.E. Bixby, V. Chvátal & W. Cook (1998) On the solution of traveling salesman problems. *Documenta Mathematica* Extra Volume ICM III 645–656.
- [2] S. Arora, M. Grigni, D. Karger, P. Klein & A. Woloszyn (1998) A polynomial-time approximation scheme for weighted planar graph TSP. In *Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 33–41.
- [3] F. Barahona & A.R. Mahjoub (1986) On the cut polytope. *Math. Program.* 36, 157–173.
- [4] S. Boyd, S. Cockburn & D. Vella (2007) On the domino-parity inequalities for the STSP. *Math. Program.* 110, 501–519.
- [5] R.D. Carr (1997) Separating clique tree and bipartition inequalities having a fixed number of handles and teeth in polynomial time. *Math. Oper. Res.* 22, 257–265.
- [6] R.D. Carr (1996) Separating over classes of TSP inequalities defined by 0 node-lifting in polynomial time. In W.H. Cunningham, S.T. McCormick & M. Queyranne (Eds.) *Integer Programming and Combinatorial Optimization V*, pp. 460–474. Lecture Notes in Computer Science 1084. Berlin: Springer.
- [7] R.D. Carr (2004) Separation algorithms for classes of STSP inequalities arising from a new STSP relaxation. *Math. Oper. Res.* 29, 80–91.
- [8] P. Chalermsook, J. Fakcharoenphol & D. Nanongkai (2004) A deterministic near-linear time algorithm for finding minimum cuts in planar graphs. In J.I. Munro (ed.) *Proceedings of the Fifteenth Annual ACM-SIAM Symposium on Discrete Algorithms*. Philadelphia: SIAM.
- [9] V. Chvátal (1973) Edmonds polytopes and weakly Hamiltonian graphs. *Math. Program.*, 5, 29–40.

- [10] W. Cook, D. Espinoza & M. Goycoolea (2005) A study of domino-parity and k -parity constraints for the TSP. In M. Jünger & V. Kaibel (eds.) *Integer Programming and Combinatorial Optimization XI*, pp. 452–467. Lecture Notes in Computer Science vol. 3509. Berlin: Springer.
- [11] G.B. Dantzig, D.R. Fulkerson & S.M. Johnson (1954) Solution of a large-scale traveling salesman problem. *Oper. Res.*, 2, 393–410.
- [12] V.G. Deineko, B. Klinz & G.J. Woeginger (2006) Exact algorithms for the hamiltonian cycle problem in planar graphs. *Oper. Res. Lett.*, 34, 269–274.
- [13] E.W. Dijkstra (1959) A note on two problems in connection with graphs. *Numerical Mathematics*, 1, 269–271.
- [14] F. Dorn, E. Penninx, H. Bodlaender & F.V. Fomin (2005) Efficient exact algorithms on planar graphs: exploiting sphere cut branch decompositions. In G.S. Brodal & S. Leonardi (eds.) *Proceedings of ESA 2005*, pp. 95–106. Lecture Notes in Computer Science vol. 3669. Berlin: Springer.
- [15] J. Edmonds (1965) Maximum matching and a polyhedron with 0-1 vertices. *J. Res. Nat. Bur. Standards*, 69B, 125–130.
- [16] L. Fleischer, A.N. Letchford & A. Lodi (2006) Polynomial-time separation of a superclass of simple comb inequalities. *Math. Oper. Res.*, 31, 696–713.
- [17] L. Fleischer & E. Tardos (1999) Separating maximally violated comb inequalities in planar graphs. *Math. Oper. Res.*, 24, 130–148.
- [18] A.V. Goldberg & R.E. Tarjan (1988) A new approach to the maximum flow problem. *J. of the ACM*, 35, 921–940.
- [19] M. Grötschel & O. Holland (1987) A cutting plane algorithm for minimum perfect 2-matching. *Computing*, 39, 327–344.
- [20] M. Grötschel, L. Lovász & A.J. Schrijver (1988) *Geometric Algorithms and Combinatorial Optimization*. Wiley: New York.
- [21] M. Grötschel & M.W. Padberg (1979) On the symmetric travelling salesman problem I: inequalities. *Math. Program.*, 16, 265–280.

- [22] M. Grötschel & M.W. Padberg (1979) On the symmetric travelling salesman problem II: lifting theorems and facets. *Math. Program.*, 16, 281–302.
- [23] M. Held & R.M. Karp (1962) A dynamic programming approach to sequencing problems. *SIAM Review*, 10, 196–210.
- [24] M.R. Henzinger, P. Klein, S. Rao & S. Subramanian (1997) Faster shortest-path algorithms for planar graphs. *J. Comput. System Sci.*, 55, 3–23.
- [25] M. Jünger, G. Reinelt, G. Rinaldi (1995) The traveling salesman problem. In M. Ball, T. Magnanti, C. Monma & G. Nemhauser (eds.) *Network Models*, pp. 225–330. Handbooks in Operations Research and Management Science vol. 7. Amsterdam: Elsevier.
- [26] M. Jünger, G. Reinelt & G. Rinaldi (1997) The traveling salesman problem. In M. Dell’Amico, F. Maffioli & S. Martello (eds.) *Annotated Bibliographies in Combinatorial Optimization*. Chichester: Wiley.
- [27] A.N. Letchford (2000) Separating a superclass of comb inequalities in planar graphs. *Math. Oper. Res.*, 25, 443–454.
- [28] A.N. Letchford & A. Lodi (2002) Polynomial-time separation of simple comb inequalities. In W.J. Cook & A.S. Schulz (eds) *Integer Programming and Combinatorial Optimization 9*. Lecture Notes in Computer Science vol 2337. Berlin: Springer.
- [29] A.N. Letchford & N.A. Pearson (2005) A fast algorithm for minimum weight odd cuts and circuits in planar graphs. *Oper. Res. Lett.*, 33, 625–628.
- [30] A.N. Letchford, G. Reinelt & D.O. Theis (2004) A faster exact separation algorithm for blossom inequalities. In G. Nemhauser & D. Bienstock (eds.) *Integer Programming and Combinatorial Optimization 10*. Lecture Notes in Computer Science, Vol. 3064. Berlin: Springer-Verlag.
- [31] R.J. Lipton & R.E. Tarjan (1979) A separator theorem for planar graphs. *SIAM J. Appl. Math.*, 36, 177–189.
- [32] D. Naddef (2002) Polyhedral theory and branch-and-cut algorithms for the symmetric TSP. In G. Gutin & A. Punnen (eds.) *The Traveling Salesman Problem and its Variations*. Kluwer Academic Publishers.

- [33] D. Naddef (2004) The domino inequalities for the symmetric traveling salesman problem. In M. Grötschel (ed.) *The Sharpest Cut: The Impact of Manfred Padberg and his Work*. MPS/SIAM series in Optimization. SIAM: Philadelphia.
- [34] D. Naddef & E. Wild (2003) The domino inequalities: new facets for the symmetric traveling salesman polytope. *Math. Program.*, 98, 223–251.
- [35] H. Nagamochi, T. Ono & T. Ibaraki (1994) Implementing an efficient minimum capacity cut algorithm. *Math. Program.*, 67, 325–341.
- [36] M.W. Padberg & M.R. Rao (1982) Odd minimum cut-sets and b -matchings. *Math. Oper. Res.*, 7, 67–80.
- [37] M.W. Padberg & G. Rinaldi (1991) A branch-and-cut algorithm for the resolution of large-scale symmetric travelling salesman problems. *SIAM Rev.*, 33, 60–100.
- [38] C.H. Papadimitriou & M. Yannakakis (1993) The traveling salesman problem with distances one and two. *Math. Oper. Res.*, 18, 1–11.
- [39] W.-K. Shih, S. Wu & Y.S. Kuo (1990) Unifying maximum cut and minimum cut of a planar graph. *IEEE Transactions on Computers*, 39, 694–697.
- [40] J.W.J. Williams (1964) Algorithm 232: Heapsort. *ACM Communications*, 7, 347–348.